Combinatorial Objects in Bio-Algorithmics: Related problems and complexities

Guillaume Blin

UNIVERSITÉ
PARIS-EST

UNIVERSITÉ PARIS-EST MARNE-LA-VALLÉE

LABORATOIRE D'INFORMATIQUE GASPARD MONGE

HABILITATION À DIRIGER DES RECHERCHES

présentée par

Guillaume BLIN

Combinatorial Objects in Bio-Algorithmics : Related problems and complexities

Soutenue publiquement le 18 Juin 2012

devant le jury composé de

Maxime Crochemore	Professeur Emerit, Université Paris-Est Marne-la-Vallée, Examinateur
Thierry Lecroq	Professeur, Université de Rouen, Examinateur
Bernard Moret	Professeur, Ecole Polytechnique Fédérale de Lausanne, Rapporteur
Eric Rivals	Directeur de Recherche CNRS, Université de Montpellier, Examinateur
Marie-France Sagot	Directeur de Recherche INRIA, Université Claude Bernard, Rapporteur
Laurent Vuillon	Professeur, Université de Savoie, Rapporteur

Contents

Remerciements

Bernard Moret, Marie-France Sagot et Laurent Vuillon m'ont honoré en acceptant d'être les rapporteurs de ce mémoire. Je les en remercie vivement.

Je remercie sincèrement Maxime Crochemore, Thierry Lecroq et Eric Rivals de m'avoir fait l'honneur de participer au jury.

Merci, tout particulièrement, à Maxime Crochemore. Sans son aide et ses conseils avisés, la thématique Algorithmique pour la Bioinformatique du LIGM n'en serait pas là.

Merci à Stéphane Vialette de m'avoir suivi dans le projet fou de faire de notre groupe une entité à visibilité internationale. Je le remercie aussi de sa patience quand dans le bureau que l'on partage je lui fais découvrir des musiques qu'il dit improbables. Il est et restera un modèle pour moi au même titre que mes directeurs de thèses l'ont été et le sont toujours (Guillaume Fertin et Irena Rusu).

Merci aux membres de la famille du LIGM d'être ce qu'ils sont. Lorsque je quitterai le LIGM, ce sera avec beaucoup de regrets.

Un énorme merci à celles qui font fonctionner le LIGM et l'IGM, avec qui les pauses cafés sont bien plus plaisantes.

Merci à mes co-auteurs qui ont indéniablement tous contribué à mon épanouissement scientifique.

Merci à Florian Sikora et Paul Morel d'avoir accepter d'être mes doctorants cobayes.

Finalement, un grand merci aux membres de ma famille et à mes amis. Tout particulièrement ma mère que j'aime et que j'adore toujours.

Introduction

I defended my PhD the 17th of November 2005. In the mean time (September 2005), I joined the Laboratoire d'Informatique Gaspard-Monge as an Assistant Professor for a one year period (as a result – at least partially – of an unformal discussion in a RER B with Maxime Crochemore). At that time, the bioinformatic research group was only composed of two Assistant Professors freshly elevated to the rank of Doctors (including myself) and three PhD students (directed by the INRIA Senior Researcher Marie-France Sagot from Lyon). Somehow, the group was leaderless since no Full Professor, nor even Associate ones were physically in the laboratory. Indeed, Maxime Crochemore (which was leading the research group) was Deputy Scientific Director of the Information and Communication Department of CNRS from 2004 to 2006 and thus was not often in the lab. In this context, my Assistant Professor period gave rise to both cutting the umbilical cord with my scientific mentors (Guillaume Fertin and Irena Rusu) and enhancing my collaborators set. To do so, I did not hesitated to migrate to Montréal, Quebec for a four weeks period just a couple of days after my PhD defense. This trip gave me the opportunity to meet some of my nowadays collaborators (Mathieu Blanchette, Nadia El-Mabrouk, Annie Chateau and Cédric Chauve) and work on new aspects and problems in the field of comparative genomics which led to two inproceedings and a journal paper. Back in France, I was contacted by one of my PhD reviewer – namely Hélène Touzet – that had nice intuitions on the links between two problems I faced during my PhD: LAPCS and EDIT (see Chapter 1 for more details). We ended up with a nice unifying framework called ALIGN bringing together most of the comparison models for arc-annotated sequences. Working without a PhD director supervision with this researchers undeniably contributed to my scientific bloom.

In September 2006, simultaneously to my access to a permanent position as Associate Professor at the LIGM, both the other Assistant Professor and the only remaining PhD student were leaving the group (for respectively an Associate Professor and a Postdoctoral postions); leaving me as the only member of the bioinformatic research group of the LIGM. I have to admit now, in retrospect, that this research confinement conjugated to the gruelling teaching work as a Junior Associate Professor were not the kind of scientific fertilizer a young researcher needs. Nevertheless, at that

time, I mostly considered my situation as a challenging one. Indeed, everything needed to be done. Considering the number of positions available each year for the laboratory, I knew that my only hope lied in both persuading a collaborator to join me as a CNRS researcher in the quest of rebuilding an entire research group and expanding my set of collabotators. Well, providence could not make a better choice than Stéphane Vialette, my favorite collaborator, who joined me as a CNRS CR1 in September 2007 (while Maxime Crochemore was becoming Professor Emeritus). From then, the research group continued to expand : a PhD student (Florian Sikora) that I co-directed with Stéphane from September 2008 to September 2011, a CNRS Director of Research (Gregory Kucherov) since January 2011, a PhD student (Paul Morel) that I co-direct with Stéphane since September 2011, and an Associate Professor (Philippe Gambette) since September 2011. Aside the expansion of the group, an achievement of our conjugated efforts with Stéphane to rebuild the group lies on the acceptance of our "ANR Jeune Chercheur" project named BIRDS (2010-2014) for which I am the coordinator. From a personal point of view, I consider my year of Temporary Full Researcher at the CNRS (2010-2011) as an achievement which allowed me to fully expressed my capabilities as a researcher. Indeed, it gave me the opportunity to spare most of my time travelling and visiting collaborators in Portugal, Italy, Germany, Canada and USA. These visits resulted in multiple results and articles (a dozen). Even, if in my proposal when I applied to this position, I stated that I will take the advantage of that period to write my HDR, I could not find the time to do so. I feel now that the time has come.

Let us now get into a more scientific description of my research. From 2003, my research activities are centred around the *biological objects comparison*. Indeed, I have investigated the algorithmic study of numerous biological problems requesting biological entities comparison. Among those entities, we can mention strings, permutations, arc-annotated sequences, 2-intervals, binary matrix, directed acyclic graphs, linear graphs, signed integer sequences, as well as trees and forests. Comparison plays a central role in the study of biological process. Indeed, very often *similarity induces common functions*. The various aspects of comparison that I have faced are the alignment, computation of parsimonious scenari, computation of distances/scores, reconstruction, prediction, motifs search. Together with the variety of definitions of *comparison*, my works have considered a large diversity of biological entities; namely RNA structures, protein structures and their interactions, genes order and inheritance, SNPs. The initial phase of each of my studies was to find or provide a suitable algorithmic framework representing the studied entity; simultaneously adapted to the desired expressiveness level and allowing benefiting from interesting algorithmic properties.

For each of the problems studied, the approach has been, in a first step, to study the classical complexity by providing, when it was possible, an exact optimal polynomial-time algorithm or by proving that such an algorithm cannot be computed (*i.e.,* **NP**-completeness theory). In the latter case, in order to provide an algorithmic solution, we considered trade-off based solutions: approximation, parameterized complexity and heuristics. Considering approximation, a compromise on the optimality of the result is done (we can guarantee a result always away from the optimal solution of a bounded distance). On the contrary, considering heuristics, we cannot provide such guarantee. We mainly ensure a suitable behavior on a test data set. The trade-off involved when considering parameterized complexity is quite different; this last relies on the class of considered instances. Indeed, the parameterized complexity is based on refocusing the combinatorial explosion of the

complexity on a parameter considered as small in practical applications. In a similar manner to classical complexity, when appoximation or parameterized solutions could not been provided, I have tried to prove that such solutions could not be derived (W[1] and **APX** hardness).

The aim of this manuscript is to exhibit my contributions in several area of Bio-Algorithmics. Rather than an exhaustive presentation of my works, I have made the choice of presenting known results (including our contributions) on a representative subset of the problems I have been involved in since 2005. Readers interesting in previous work may refer to my PhD. For ease of readability, afterwards, I will regroup the results obtained according to the biological problems: i) RNA structures comparison, ii) Genomes comparison and iii) Pattern matching in biological networks and their respective combinatorial objects: i) Arc-annotated sequences, ii) Permutations and Sequences and iii) Graphs. For the prerequisites, the reader is expected to be familiar with basic graph theory, classical complexity theory and parameterized complexity theory.

More precisely, Chapter 1 will be devoted to the Arc-Annotated Sequences that are used in RNA structure comparison. We will focus on five problems that we investigated: LAPCS, APS, MAPCS, EDIT and ALIGN. In Chapter 2, we will consider the two main research area related to comparative genomics we were involved in: Gene clusters detection and (dis)similarity measures computation – which rely on permutation and string representations. In Chapter 3, we will present some results that were obtained mainly during the PhD of Florian Sikora.

1

RNA structure comparison: Arc-Annotated Sequences

Contents

1.1 Introduction

Structure comparison for RNA has become a central computational problem bearing many computer science challenging questions. Indeed, RNA secondary structure comparison is essential for

- (i) the identification of highly conserved structures during evolution (which cannot always be detected in the primary sequence, since it is often not preserved) which suggest a significant common function for the studied RNA molecules,
- (ii) RNA classification of various species (phylogeny),
- (iii) RNA folding prediction by considering a set of already known secondary structures and
- (iv) the identification of a consensus structure and consequently of a common role for molecules.

From an algorithmic point of view, RNA structure comparison was first considered in the framework of ordered trees by [116]. A decade after, it was also considered in the framework of *arc-annotated sequences* by [68]. An arc-annotated sequence is a pair (S, P) where S is a sequence of RNA bases and P represents hydrogen bonds between pairs of elements of S. From a purely combinatorial point of view, arc-annotated sequences are a natural extension of sequences. However, using arcs for modeling non-sequential information together with restrictions on the relative positioning of arcs allow for varying restrictions on the structure of arc-annotated sequences.

Different pattern matching and motif search problems have been considered in the context of arc-annotated sequences among which we can mention the Longest Arc-Annotated Subsequence (LAPCS) problem, the Arc Preserving Subsequence (APS) problem, the Maximum Arc-Preserving Common Subsequence (MAPCS) problem, the Edit-distance for arc-annotated sequence (EDIT) problem and the unifying ALIGN hierarchy.

This chapter is devoted to presenting algorithmic results for these arc-annotated problems. Note that, in the field of RNA comparison, we also got some results [35; 38] on the 2-interval framework introduced by [126; 127]. Since these results were obtained during the PhD and not further investigated since then, they are thus not presented in this manuscript (nor the 2-interval framework).

1.2 Preliminaries

1.2.1 Arc-annotated sequences

Given a finite alphabet Σ, an arc-annotated sequence is defined by a pair (S, P), where S is a string of Σ^* and P is a set of arcs connecting pairs of characters of S. The set P is usually represented by set of pairs of positions in S. Characters that are not incident to any arc are called *free*.

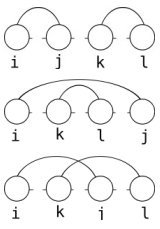
In the context of RNA structures, we have $\Sigma = \{A, C, G, U\}$, and S and P represent the nucleotide sequence and the hydrogen bonds of the RNA structure, respectively. Characters in S are thus often referred to as *bases*.

Relative positioning of arcs is of particular importance for arc-annotated sequences and is completely described by three binary relations. Let $p_1 = (i, j)$ and $p_2 = (k, l)$ be two arcs in P that do not share a vertex. Define

the precedence relation ($<$) : $p_1 < p_2$ if $i < j < k < l$

the embedding relation ($\sqsubset$) : $p_1 \sqsubset p_2$ if $i < k < l < j$

the crossing relation ($\between$) : $p_1 \between p_2$ if $i < k < j < l$



Remind that, using arcs for modeling non-sequential information together with these relations allow us for varying restrictions on the complexity of arc-annotated sequences.

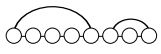
1.2.2 Hierarchy

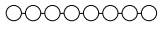
Five levels of arc structure have been initially considered in the pioneer work of [67]:

UNLIMITED (UNLIM) – *no restriction at all,* 

CROSSING (CROSS) – *there is no character incident to more than one arc,* 

NESTED (NEST) – *there is no character incident to more than one arc and no arcs are crossing,* 

CHAIN (CHAIN) – *there is no character incident to more than one arc, no arcs are crossing and no arc embedded into another, and* 

PLAIN – *there is no arc.* 

The induced hierarchy is described by the following chain of inclusion:

$$\text{PLAIN} \subset \text{CHAIN} \subset \text{NESTED} \subset \text{CROSSING} \subset \text{UNLIMITED}.$$

1.2.3 Refined Hierarchy

[77] extended the above-mentioned hierarchy by introducing a new refinement of the NESTED level called STEM: no character is incident to more than one arc, and given any two arcs, one is embedded into the other.

For providing a unified framework and a better understanding of the inner complexity of the problems related to arc-annotated sequences, with G. Fertin, R. Rizzi and S. Vialette, we [31] proposed to further refine the hierarchy following the example of [126; 127] in the context of 2-*intervals* (a simple abstract structure for modeling RNA secondary structures). The refinement consists in splitting those models of arc-annotated sequences into more precise relations between arcs, taking advantage of the combinatorics induced by the relations $<$, $\sqsubset$, and $\between$.

Two arcs p_1 and p_2 are R-comparable for some $R \in \{<, \sqsubset, \between\}$ if $p_1 R p_2$ or $p_2 R p_1$. Let P be a set of arcs and $\mathcal{R}$ be a non-empty subset of $\{<, \sqsubset, \between\}$. The set P is said to be $\mathcal{R}$-*comparable* if any two distinct arcs of P are R-comparable for some $R \in \mathcal{R}$. An arc-annotated sequence (S, P) is said to be an $\mathcal{R}$-arc-annotated sequence for some non-empty subset $\mathcal{R} \subseteq \{<, \sqsubset, \between\}$ if P is $\mathcal{R}$-comparable. By abuse of notation, we will write $R = \emptyset$ in case $P = \emptyset$.

As a straightforward illustration of the above definitions, most levels in the classical hierarchy can be expressed in terms of a combination of the three relations: PLAIN is fully described by $\mathcal{R} = \emptyset$, CHAIN is fully described by $\mathcal{R} = \{<\}$, STEM is fully described by $\mathcal{R} = \{\sqsubset\}$, NESTED is fully described by $\mathcal{R} = \{<, \sqsubset\}$ and CROSSING is fully described by $\mathcal{R} = \{<, \sqsubset, \between\}$. The key point is to observe that this refinement allows us to consider new levels for arc-annotated sequences, namely $\mathcal{R} = \{\between\}$, $\mathcal{R} = \{<, \between\}$ and $\mathcal{R} = \{\sqsubset, \between\}$.

1.2.4 Alignment

Given two sequences S and T on a common alphabet Σ, we define an *alignment* of S and T as a pair of sequences (S', T') built from S and T on $\Sigma \cup \{-\}$ ($-$ is usually referred to as a *gap*) such that (i) $|S'| = |T'|$, (ii) for any $1 \leq i \leq |S'|$, either $S'[i] = T'[i] \neq -$ or exactly one of $S'[i]$ and $T'[i]$ is a gap, and (iii) removing the gaps from S' (resp. T') yields S (resp. T).

Let (S', T') be an alignment of S and T. For any $1 \leq i \leq |S'|$ such that $S'[i] \neq -$, character $S'[i]$ is said to be *aligned* with character $T'[i]$ if $T'[i] \neq -$, and *deleted* otherwise. Similarly, For any $1 \leq i \leq |T'|$ such that $T'[i] \neq -$, character $T'[i]$ is said to be *aligned* with character $S'[i]$ if $S'[i] \neq -$, and *inserted* otherwise. An illustration is given in Figure 1.1.

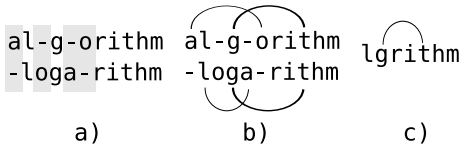


Figure 1.1: Illustration of a) sequences alignment leading to a common subsequence which is "lgrtihm", b) an arc-preserving alignment of two arc-annotated sequences and c) the resulting common arc-annotated subsequence of b)

An alignment (S', T') of two arc-annotated sequences (S, P) and (T, Q) is *arc-preserving* if the arcs induced by (S', T') are preserved, *i.e.*, the arcs induced by the aligned bases are preserved. In this context, the notion of common subsequence is extended by including the common arcs – that is the arcs that have been preserved by the alignment.

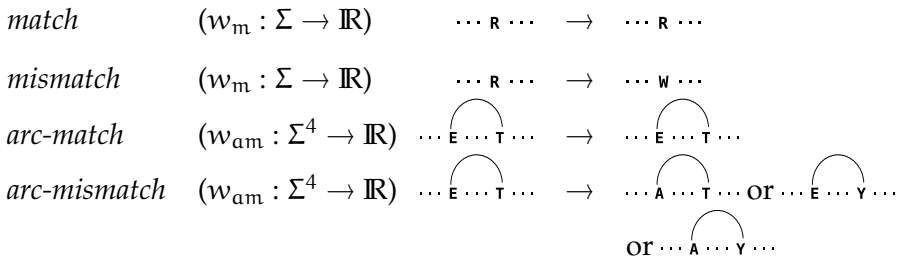
1.2.5 Edit Operations

Following the example of stringology, when comparing two arc-annotated sequences (S, P) and (T, Q), instead of computing an alignment, one might consider a set of edit operations (together with their associate costs) that alter arc-annotated sequences, and seek for a minimal cost sequence according to these operations that leeds from (S, P) to (T, Q).

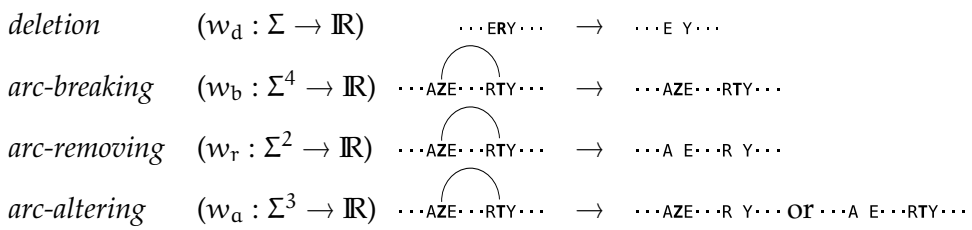
Formally, given a set of edit operations $\mathcal{E}$ and two arc-annotated sequences (S, P) and (T, Q), an *edit-script* from (S, P) to (T, Q) refers to a series of non-oriented operations of $\mathcal{E}$ transforming (S, P) into (T, Q). The *cost of an edit-script* from (S, P) to (T, Q), denoted $\mathtt{cost}((S, P), (T, Q), \mathcal{E})$, is the sum of the costs of all operations involved in the edit-script. The *edit-distance* between (S, P) and (T, Q) is the minimum cost of an edit-script from (S, P) to (T, Q).

The classical approach is to consider a subset of the operations introduced by [86] which can be divided into two groups:

Substitution operations, inducing renaming of characters in the arc-annotated sequence:



Deletion operations, inducing deletion of characters and/or of arcs in the arc-annotated sequence:



1.3 Longest Arc-Preserving Common Subsequence

1.3.1 Definition

The LAPCS problem has been introduced by [67] and is defined as follows: given two arc-annotated sequences (S, P) and (T, Q), find an arc-preserving common subsequence of maximal length. The computational complexity of the LAPCS problem has been studied in [67; 68; 88; 95; 79; 39], and the main results are summarized in tables 1.1, 1.2 and 1.3.

In the sequel, we use the notation LAPCS(A, B) to represent the LAPCS problem where the arc structure of S (resp. T) – namely P (resp. Q) – is of level A (resp. B).

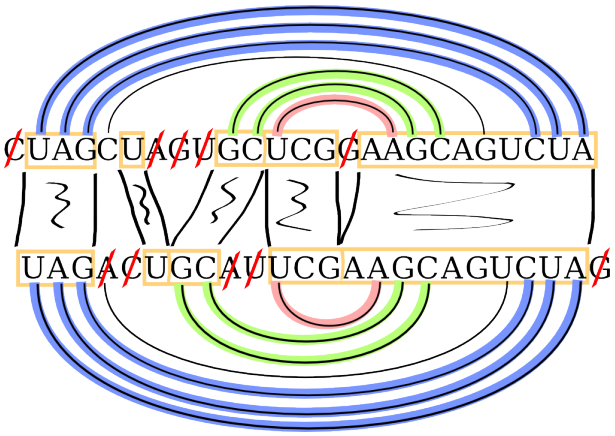


Figure 1.2: Illustration of LAPCS

1.3.2 Classical complexity

[67] proved that LAPCS(CHAIN, CHAIN) is polynomial-time solvable, whereas both LAPCS(UNLIMITED, PLAIN) and LAPCS(CROSSING, PLAIN) are **NP**-complete (reductions from Independent Set). [95] proved that LAPCS(NESTED, NESTED) is **NP**-complete (reduction from Independent Set). Complementing these results, [88] designed an $O(nm^3)$ time algorithm for LAPCS(NESTED, CHAIN) and LAPCS(CHAIN, CHAIN). Recently, [39] proved that LAPCS(STEM, STEM) is **NP**-complete (reduction from 3-SAT).

A × B	LAPCS
STEM × STEM	**NP**-complete – [39]
CHAIN × CHAIN	$O(nm^3)$ – [88]
NEST × CHAIN	
NEST × NEST	**NP**-complete even for unary, c-fragment (with $c > 2$) and c-diagonal (with $c > 1$) – [95]
CROSS × CHAIN CROSS × NEST	**NP**-complete – [67]
CROSS × CROSS	**NP**-complete – [67] but polynomial-time solvable for 1-fragment LAPCS(CROSSING, CROSSING) and 0-diagonal LAPCS(CROSSING, CROSSING) [95]
UNLIM × CHAIN	**NP**-complete – [67]
UNLIM × NEST	
UNLIM × CROSS	
UNLIM × UNLIM	

Table 1.1: LAPCS classical complexity with $n = |S|$ and $m = |T|$

[95] further investigated the problem by studying restricted cases: namely, c-FRAGMENTED, c-DIAGONAL and UNARY LAPCS(NESTED, NESTED). Given two arc-annotated sequences which are divided into fragments of lengths exactly c (the last fragment can have a length less than c), the *c-fragment* LAPCS problem with $c \geq 1$, is defined as the classical LAPCS problem with the extra constraint that the allowed matches are those between fragments at the same location [79]. The *c-diagonal* LAPCS problem with $c \geq 0$ is an extension of c-fragment LAPCS, where character $S[i]$ is allowed only to match a character in the range $T[i - c, i + c]$. The UNARY subproblem considers arc-annotated sequences based on a single character alphabet. [95] showed the **NP**-hardness of the c-fragment (with $c > 2$) and c-diagonal (with $c > 1$) LAPCS (NESTED,NESTED problem. They also proved that the 1-fragment LAPCS(CROSSING, CROSSING) and 0-diagonal LAPCS(CROSSING, CROSSING) are solvable in $O(n)$ time.

1.3.3 Parameterized complexity

Considering the parameter l as being the desired length of common subsequence, [67], by using one of the previous above-mentioned reduction for LAPCS(UNLIMITED, PLAIN) and by providing a reduction from CLIQUE to LAPCS(CROSSING, CROSSING), proved that both LAPCS(UNLIMITED, PLAIN) and LAPCS(CROSSING, CROSSING) are **W[1]**-complete when parameterized by l. Moreover, [68] proved that whereas LAPCS(CROSSING, CROSSING) is **W[1]**-complete, the prob-

A × B			LAPCS
STEM	×	STEM	FPT when parameterized by the number of deletion – [1]
NEST	×	CHAIN	FPT when parameterized by the bandwidth or the nesting depth – [67], FPT when parameterized by the number of deletion – [1]
NEST	×	NEST	
CROSS	×	CHAIN	FPT when parameterized by the bandwidth or the cutwidth – [67], [87]
CROSS	×	NEST	
CROSS	×	CROSS	**W**[**1**]-complete and FPT when parameterized by the bandwidth or the cutwidth – [67], FPT when parameterized by the desired common subsequence length – [1]
UNLIM	×	CHAIN	**W**[**1**]-complete – [67]
UNLIM	×	NEST	
UNLIM	×	CROSS	
UNLIM	×	UNLIM	

Table 1.2: LAPCS parameterized complexity with $n = |S|$ and $m = |T|$.

lem becomes fixed-parameter tractable when parameterized by the arc cutwidth. The *arc cutwidth* [68] of an arc-annotated sequence is defined as the maximal number of arcs that cross or end at any arbitrary position of the sequence. If both sequences have their cutwidth bounded by some k, the problem, as shown by Evans, can be solved in $O(9^k nm)$ time, where $|S| = n$ and $|T| = m$. Evans also investigated the parameterized complexity of the problem considering two other parameters: the bandwidth and the nesting depth. The *bandwidth* d of an arc-annotated sequence (S, P) is defined by $max_{(i,j)\in P}\{|j - i|\}$ and its *nesting depth* s is equal to $max\{|P'|\}$, where $P' \subseteq P$ is a set of pairwise nested arcs. Evans showed that, if both sequences have their nesting depth bounded by some s, LAPCS(NESTED, NESTED) can be solved in $O(s^2 4^s nm)$ time, where $|S| = n$ and $|T| = m$. In case the arcs do not share endpoints, both cutwidth and nesting depth are always no more than bandwidth. Thus, Evans, was able to extend the previously mentioned results to the parameter d. Finally, one has to observe that if the bandwidth of the arc-structure is bounded by a logarithm of the maximal sequence length n, LAPCS can be solved in $O(n^2 m)$ time even for CROSSING type arc structures. Moreover, since the cutwidth is equal to 1 in the case of LAPCS(CHAIN, CHAIN), one can use the algorithm for LAPCS(CROSSING, CROSSING) to solve this problem in $O(nm)$ time.

For LAPCS(NESTED, NESTED), [1] designed an algorithm which determines in time $O(3.31^{k_1+k_2} n)$ whether an arc-preserving common subsequence can be obtained by deleting (together with incident arcs) k_1 characters from S and k_2 from T, thereby proving that LAPCS(NESTED, NESTED) is fixed-parameter tractable when parameterized by the number of deletions. Finally, [1] showed that c-fragment LAPCS(CROSSING, CROSSING) and c-diagonal LAPCS(CROSSING, CROSSING)

A × B	LAPCS
NEST × CHAIN	2-approximable – [87], PTAS for
NEST × NEST	c-fragmented and c-diagonal cases – [95]
CROSS × CHAIN	**MaxSNP**-hard, 2-approximable – [87]
CROSS × NEST	
CROSS × CROSS	
UNLIM × CHAIN	Cannot be approximated within ratio n^{ϵ} for any $\epsilon \in (0, \frac{1}{4})$ – [87]
UNLIM × NEST	
UNLIM × CROSS	
UNLIM × UNLIM	

Table 1.3: LAPCS approximability.

parameterized by the length l of the desired common subsequence are solvable in $O((B+1)^l B + c^3 n)$ time, with $B = c^2 + 2c - 1$ and $B = 2c^2 + 7c + 2$, respectively.

1.3.4 Approximability

[87] proved that LAPCS(CROSSING, CROSSING) admits a simple 2-approximation algorithm running in $O(nm)$ time whereas LAPCS(UNLIMITED, PLAIN) cannot be approximated within ratio n^{ϵ} for any $\epsilon \in (0, \frac{1}{4})$, where n denotes the length of the longest input sequence. In the same paper, they proved that LAPCS(CROSSING, PLAIN) is **MaxSNP**-hard, thereby excluding a polynomial-time approximation scheme (PTAS). [95] proved that both c-fragmented and c-diagonal LAPCS(NESTED, NESTED) have a PTAS. They also gave a $\frac{4}{3}$-approximation algorithm for the unary LAPCS(NESTED, NESTED) problem.

1.4 Arc-Preserving Subsequence

1.4.1 Definition

The APS problem is a decision problem derived from LAPCS. Given two arc-annotated sequences (S, P) and (T, Q), the APS problem asks whether (T, Q) is the LAPCS of (S, P) and (T, Q) , *i.e.*, (T, Q) is an arc-preserving subsequence of (S, P). The computational complexity of the APS problem has been studied in [67; 75; 76; 79; 32; 31], and the main results are summarized in tables 1.4 and 1.5.

In the following, we use the notation APS(A, B) to represent the APS problem where the arc structure of S (resp. T) – namely P (resp. Q) – is of level A (resp. B).

A × B	APS
CHAIN × CHAIN	O(nm) – [75; 76]
NEST × CHAIN	
NEST × NEST	
CROSS × PLAIN	**NP**-complete – [32; 31]
CROSS × CHAIN	**NP**-complete – [75; 76]
CROSS × NEST	
CROSS × CROSS	**NP**-complete – [67]
UNLIM × CHAIN	
UNLIM × NEST	
UNLIM × CROSS	
UNLIM × UNLIM	

Table 1.4: APS classical complexity with $n = |S|$ and $m = |T|$.

1.4.2 Classical complexity

[79] proved that the APS(CROSSING, CHAIN) problem is **NP**-hard from a reduction of INDEPENDENT SET. [75; 76] observed that the **NP**-completeness of the APS(CROSSING, CROSSING) and APS(UNLIMITED, PLAIN) easily follows from the work of Evans [67]. Furthermore, they gave an O(nm) time algorithm for the APS(NESTED, NESTED) problem. This algorithm can be applied to easier problems such as APS(NESTED, CHAIN), APS(NESTED, PLAIN), APS(CHAIN, CHAIN) and APS(CHAIN, PLAIN). [75; 76] mentioned that APS(CH- AIN, PLAIN) can be solved in O(n + m) time. Finally, with G. Fertin, R. Rizzi and S. Vialette, we [32; 31] proved APS(CROSSING, PLAIN) to be **NP**-complete (reduction from 3-SAT).

1.4.3 Classical complexity for the refined hierarchy

For the refined hierarchy we introduced in [32; 31], the number of complexity levels rises from 4 (not taking into account the UNLIMITED case) to 8.

On the positive side, [75; 76] have shown that APS(NESTED, NESTED) is solvable in O(nm) time. Another way of stating this result is to say that APS($\{<, \sqsubset\}, \{<, \sqsubset\}$) is solvable in O(mn) time. According to the properties of the refined hierarchy, that result may be summarized by saying that APS(R_1, R_2) for any compatible R_1 and R_2 such that $\between \notin R_1$ and $\between \notin R_2$ is polynomial-time solvable.

Conversely, the **NP**-completeness of APS(CROSSING, CROSSING) has been proved by [67]. A simple reading shows that the corresponding proof is actually concerned with $\{<, \sqsubset, \between\}$-arc-annotated sequences, and hence actually proves that APS($\{<, \sqsubset, \between\}, \{<, \sqsubset, \between\}$) is **NP**-complete. Similarly, proving that APS(CROSSING, CHAIN) is **NP**-complete, [79] actually proved that APS({<

A × B			APS
$\{<\}$	×	$\emptyset$	$O(n+m)$ [75]
$\{<\}$	×	$\{<\}$	[75; 76]
$\{\sqsubset\}$	×	$*$	
$\{<,\sqsubset\}$	×	$*$	
$\{\wr\}$	×	$\emptyset$	$O(nm^2)$ – [32; 31]
$\{\wr\}$	×	$\{\wr\}$	
$\{<,\wr\}$	×	$*$	**NP**-complete– [32; 31], [79], [67]
$\{\sqsubset,\wr\}$	×	$*$	
$\{<,\sqsubset,\wr\}$	×	$*$	

Table 1.5: APS classical refined complexity where $n = |S|$ and $m = |T|$.

$,\sqsubset,\wr\},\{<\})$ is **NP**-complete. Therefore, both APS$(\{<,\sqsubset,\wr\},\{<,\sqsubset\})$ and APS$(\{<,\sqsubset,\wr\},\{<,\wr\})$ are **NP**-complete.

With G. Fertin, R. Rizzi and S. Vialette, we [32; 31] proved that both APS$(\{\sqsubset,\wr\},\emptyset)$ and APS$(\{<$ $,\wr\},\emptyset)$ are **NP**-complete. We also gave a polynomial-time algorithm to show that both APS$(\{\wr\},\{\wr\})$ and APS$(\{\wr\},\emptyset)$ problems can be solved in $O(nm^2)$ time. In other words, we proved that the relation $\wr$ alone does not imply hardness.

The refinement that we suggested in [32; 31] shows that APS problem becomes hard when one considers sequences containing $\{\wr, R\}$-comparable for some $R \subseteq \{<,\sqsubset,\wr\}$. Therefore, crossing arcs alone do not imply APS hardness.
This remarks motivates the challenging problem of further exploring the complexity of the APS problem, and especially the parameterized views, by considering additional parameters such as the cutwidth or the depth of the arc structures.
As far as we know, this problem is completely open.

1.5 Maximum Arc-Preserving Common Subsequence

1.5.1 Definition

With G. Fertin, G. Herry and S. Vialette, we [28] introduced the MAPCS problem as an intermediate model for comparing arc-annotated sequences – lying between LAPCS and the EDIT (see Section 1.6). The MAPCS problem is defined as follows: given two arc-annotated sequences (S, P) and (T, Q), and two functions $f_b : \Sigma \to \mathbb{N}^*$ and $f_a : \Sigma^2 \to \mathbb{N}^*$, find a common arc-annotated subsequence (U, R) that maximizes the following score function: $\sum_{c \in U} f_b(c) + \sum_{(c_1,c_2) \in R} f_a(c_1, c_2)$. In other words, the MAPCS problem seeks for a common subsequence whose score takes into account both the number of bases and arcs. The computational complexity of the MAPCS problem was fully determined in [28], and the main results are summarized in Table 1.6.

In the following, we use the notation MAPCS(A, B) to represent the MAPCS problem where the arc structure of S (resp. T) – namely P (resp. Q) – is of level A (resp. B).

1.5.2 Classical complexity

With G. Fertin, G. Herry and S. Vialette, we [28], we first investigated two special cases of MAPCS, namely when one allows function f_a or f_b to return zero. We easily noticed that $f_a(x, y) = 0$ for all $(x, y) \in \Sigma^2$ reduces to the LAPCS problem. We investigated the case $f_b(x) = 0$ for all $x \in \Sigma$, problem called MAPCS$*$, and proved that MAPCS$*$(CHAIN,CHAIN) can be solved in $O(nm)$ time, MAPCS$*$(NESTED,NESTED) in $O(n^2m^2)$ time , MAPCS$*$(NESTED,CHAIN) in $O(nm^2)$ time and MAPCS$*$(UNLIMITED,NESTED) in $O(n^4 \log^3 n)$ time, where $n = |S|$ and $m = |T|$. We also proved that MAPCS$*$(CROSSING,CROSSING) is **NP**-complete by providing a reduction from CLIQUE.

We also fully investigated the complexity of MAPCS by giving an $O(nm)$ (resp. $O(nm^3)$) time algorithm for MAPCS (CHAIN,CHAIN) (resp. MAPCS (NEST,CHAIN)), and by proving that both MAPCS (NESTED,NESTED) and MAPCS (CROSSING,PLAIN) are **NP**-complete.

As far as we know, neither the parameterized complexity nor the approximability of MAPCS have been studied (except for the case where $f_a(c) = 0$, for all c, since it corresponds to LAPCS problem and inherits all its complexity results).

1.6 EDIT Distance

1.6.1 Definition

Given two arc-annotated sequences, the EDIT problem is to find the edit-distance between (S, P) and (T, Q). It has been extensively studied [86; 97; 77; 49; 33; 28; 24; 39].

A × B	MAPCS∗	MAPCS
CHAIN × CHAIN	$O(nm)$	$O(nm)$
NEST × CHAIN	$O(n^2m)$	$O(nm^3)$
NEST × NEST	$O(n^2m^2)$	**NP**-complete
CROSS × CHAIN	$O(n^4 \log^3 n)$	
CROSS × NEST		
CROSS × CROSS	**NP**-complete	
UNLIM × CHAIN	$O(n^4 \log^3 n)$	
UNLIM × NEST		
UNLIM × CROSS	**NP**-complete	
UNLIM × UNLIM		

Table 1.6: MAPCS∗ and MAPCS classical complexity for $n = |S|$ and $m = |T|$.

1.6.2 Classical complexity

[97] proved that the problem EDIT(CROSSING, PLAIN) is **NP**-complete, and gave a polynomial-time dynamic programming algorithm for the EDIT(NESTED, PLAIN) problem. [112] had previously solved EDIT(PLAIN, PLAIN).

With H. Touzet, we [49] proved that the LAPCS problem can be seen as a special case of the EDIT problem. More precisely, any edit script of minimum cost goes through a common subsequence of optimal score. This means that finding one allows to find the other. Thus, LAPCS can be seen as a particular case of EDIT where the cost system for edit operations is the following: $w_r = 2w_d = 2w_a$, and all substitution operations and arc-breakings are prohibited by an arbitrary high cost. The main idea is to penalize deletion operations proportionally to the number of bases that are deleted. This result proved that the complexity of EDIT(NESTED, NESTED) simply follows from the complexity of LAPCS(NESTED, NESTED). With G. Fertin, I. Rusu and C. Sinoquet, we [33] extended this results by showing that only a very restricted number of instances of EDIT(NESTED, NESTED) were shown to be **NP**-complete and that the corresponding cost system needed to satisfy restrictions which can be biologically discussed. Therefore, as another step towards establishing the precise complexity landscape of the EDIT problem, we considered a more accurate class of instances – but not overlapping with the one used in the proof from LAPCS –, for determining more precisely what makes the problem hard.

[77] introduced the notion of *conservative edit distance and mapping* between two RNA stem-loops in order to design a polynomial-time algorithm for comparing general secondary RNA structures using the full set of biological edit operations introduced in [86]. This algorithm is based on a decomposition in stem-loop-like substructures that are pairwised compared and used to compare complete RNA secondary structures. As mentioned in [77], whereas in the very restrictive case of

A × B			EDIT
CHAIN	×	CHAIN	$O(nm^3)$ – [97]
STEM	×	STEM	**NP**-complete – [39]
NEST	×	CHAIN	$O(nm^3)$ – [97]
NEST	×	NEST	**NP**-complete – [86] and [33]
CROSS	×	CHAIN	**NP**-complete – [97]
CROSS	×	NEST	
CROSS	×	CROSS	
UNLIM	×	CHAIN	
UNLIM	×	NEST	
UNLIM	×	CROSS	
UNLIM	×	UNLIM	

Table 1.7: EDIT classical complexity for $n = |S|$ and $m = |T|$.

conservative distance and mapping, the computation of the general edit distance is polynomial-time solvable, it was not known if the general, *i.e.*, not conservative, edit distance between two stem-loops can be also computed in polynomial-time. With S. Hamel and S. Vialette, we [39] proved that the general edit distance is indeed **NP**-complete.

1.6.3 Approximability

[97] proved that the problem EDIT(CROSSING, PLAIN) is **MaxSNP**-hard. They also shown that EDIT(NESTED, NESTED) has a polynomial-time approximation algorithm with ratio $\beta = \max\{\frac{2w_a}{w_b+w_r}, \frac{w_b+w_r}{2w_a}\}$.

1.7 ALIGN Hierarchy

1.7.1 Definition

In [49], with H. Touzet, we proposed a unifying framework – the so-called ALIGN hierarchy – that brings together most of the comparison models for arc-annotated sequences, and leads to the introduction of new comparison models that are biologically relevant. The ALIGN hierarchy model considers three edit models based on the edit operations previously introduced:

I : all substitution operations, base-deletions and arc-removings are allowed,
II : the operations of model I and arc-alterings are allowed,
and III : the operations of model II and arc-breakings are allowed.

A × B			EDIT
NEST	×	NEST	$\max\{\frac{2w_a}{w_b+w_r}, \frac{w_b+w_r}{2w_a}\}$-approximable – [97]
CROSS	×	CHAIN	**MaxSNP**-hard – [97]
CROSS	×	NEST	
CROSS	×	CROSS	
UNLIM	×	CHAIN	
UNLIM	×	NEST	
UNLIM	×	CROSS	
UNLIM	×	UNLIM	

Table 1.8: EDIT approximability for $n = |S|$ and $m = |T|$.

In order to bring together most of the comparison models for arc-annotated sequences, the ALIGN hierarchy uses generalization of the notions previously mentioned in Section 1.2. Given two arc-annotated sequences (S, P) and (T, Q), a K-*edit script* from (S, P) to (T, Q) will refer to an edit-script from (S, P) to (T, Q) only using operations of the model K. The associated cost is denoted as $\texttt{cost}(u, v, K)$. We also refine the notion of edit distance by introducing the K-*edit distance* between (S, P) and (T, Q) as the minimum cost of a K-edit script from (S, P) to (T, Q). Finding this K-edit distance is referred afterwards as the $\text{EDIT}((S, P), (T, Q), K)$ problem. Note that the specific case of $\text{EDIT}((S, P), (T, Q), \text{III})$ fully corresponds to the problem presented in the previous section.

For each model $K \in \{I, II, III\}$, let us also define an *ordering relation* $\trianglelefteq_K$: if (S, P) can be obtained from (T, Q) by a series of deletion and substitution operations of the model K, then $(S, P) \trianglelefteq_K (T, Q)$. Provided with these notations, as proposed in [49], we can extend the notion of subsequence on strings to arc-annotated sequences as follows.

***Definition* 1.7.1** (K-subsequence) Given two arc-annotated sequences (S, P) and (T, Q), and an edit model $K \in \{I, II, III\}$, (S, P) is said to be a K-subsequence of (T, Q) if, and only if, $(S, P) \trianglelefteq_K (T, Q)$.

Given three arc-annotated sequences (S, P), (T, Q) and (U, R) such that $(U, R) \trianglelefteq_K (S, P)$ and $(U, R) \trianglelefteq_K (T, Q)$, (U, R) is said to be a *common* K-*subsequence* of (S, P) and (T, Q). We define the cost of a common K-subsequence (U, R) of (S, P) and (T, Q) as the minimum sum of operation costs needed to transform (S, P) into (U, R) and (T, Q) into (U, R): $\texttt{cost}((S, P), (U, R), K) + \texttt{cost}((T, Q), (U, R), K)$.

When dealing with plain sequences, it is well-known that each edit script can be associated with a common subsequence of the same cost. In [49], we proved that this property is still valid with K-edit scripts on arc-annotated sequences. Namely, that given two arc-annotated sequences (S, P) and (T, Q), and an edit model $K \in \{I, II, III\}$, solving the $\text{EDIT}((S, P), (T, Q), K)$ problem is equivalent to finding a common K-subsequence of (S, P) and (T, Q) of minimal cost. This property allowed us to shed new light on the link between LAPCS and $\text{EDIT}((S, P), (T, Q), \text{III})$ models.

Let us now present the ALIGN hierarchy that "simply" considers K-supersequences instead of K-subsequences. We will see that this alternative point of view is a fruitful perspective and brings

new insights on arc-annotated comparison.

***Definition* 1.7.2** (K-supersequence) Given two arc-annotated sequences (S, P) and (T, Q), and an edit model $K \in \{I, II, III\}$, (S, P) is said to be a K-supersequence of (T, Q) if, and only if, $(T, Q) \trianglelefteq_K (S, P)$.

In a similar way as for common subsequences, given three arc-annotated sequences (S, P), (T, Q) and (U, R), (U, R) is a *common* K-*supersequence* of (S, P) and (T, Q) if $(S, P) \trianglelefteq_K (U, R)$ and $(T, Q) \trianglelefteq_K (U, R)$. The cost of (U, R) is defined as $\texttt{cost}((U, R), (S, P), K) + \texttt{cost}((U, R), (S, P), K)$. In [49], we proved that given two arc-annotated sequences (S, P) and (T, Q), and an edit model $K \in \{I, II, III\}$, there exists a common K-subsequence of (S, P) and (T, Q) of cost α iff there exists a common K-supersequence of (S, P) and (T, Q) of the same cost. Roughly, we proved that each EDIT problem can reduce to finding an optimal supersequence.

It is worth to notice here is that the type of the common supersequence is not guaranteed to be the same as the type of the common subsequence. Indeed, as we noticed in [49], when constructing the set of arcs of the common K-supersequence, it is likely to create crossing arcs or multiple arcs incident to a single character that are absent in the initial sequences. Thus, in general, for arc-annotated sequences of a given type, searching for a common supersequence of the same type is more restrictive than searching for a common subsequence. This is the foundation of the ALIGN *hierarchy*.

***Definition* 1.7.3** (Arc-annotated sequence alignment) Given three types of sequences A, B and C of {NESTED, CROSSING, UNLIMITED} and an edit model $K \in \{I, II, III\}$, the ALIGN$(A, B, K) \rightarrow C$ problem is defined as:

INPUT: two arc-annotated sequences (S, P) and (T, Q) of type A and B respectively.
OUTPUT: a common K-supersequence (U, R) of type C of minimum cost.

1.7.2 Classical complexity

Since ALIGN$(A, B, K) \rightarrow C$ is equivalent to ALIGN$(B, A, K) \rightarrow C$, we can always assume that $B \subseteq A$. Moreover, in order for the problem to be meaningful, we impose $A \subseteq C$. Therefore, the hierarchy contains 30 entries when considering all relevant possibilities for A, B, C and K.

The first result of interest is that given two types A, B in {NESTED, CROSSING, UNLIMITED} and an edit model $K \in \{I, II, III\}$, the EDIT(A, B, K) and ALIGN$(A, B, K) \rightarrow$ UNLIM problems are equivalent. We now have a closer look at each edit model.

Ordered trees and the edit model I

In [49], we stated that comparing arc-annotated sequences of NESTED types using the edit model I amounts to comparing ordered trees. Indeed, each pair of connected bases corresponds to an internal node, and each single base corresponds to a leaf. Moreover, in this model, considering arc-annotated I-supersequences of UNLIMITED type is meaningless. Indeed, first, note that given two types A, B in {NEST, CROSS}, the ALIGN$(A, B, I) \rightarrow$ UNLIM and ALIGN$(A, B, I) \rightarrow$ CROSS problems are equivalent. Moreover, given a type B in {NEST, CROSS}, the ALIGN$(\text{UNLIM}, B, I) \rightarrow$ UNLIM problem has the same complexity as ALIGN$(\text{CROSS}, B, I) \rightarrow$ CROSS.

A × B → C						EDIT	model I
NEST	×	NEST	→	NEST			$O(n^4)$ – [89]
NEST NEST	× ×	NEST NEST	→ →	CROSS UNLIM		×	$O(n^3 \log(n))$ – [92]
CROSS CROSS	× ×	NEST NEST	→ →	CROSS UNLIM		×	$O(n^3 \log(n))$ – [98]
CROSS CROSS	× ×	CROSS CROSS	→ →	CROSS UNLIM		×	**NP**-complete – [98]
UNLIM	×	NEST	→	UNLIM		×	$O(n^3 \log(n))$ – [49]
UNLIM	×	CROSS	→	UNLIM		×	**NP**-complete – Ma [98]
UNLIM	×	UNLIM	→	UNLIM		×	**NP**-complete – Ma [98]

Table 1.9: ALIGN hierarchy for the edit model I. Complexity results are indicated for two arc-annotated sequences (S, P) and (T, Q) s.t. $\max(|S|, |T|) = n$.

Consequently, 9 out of 10 entries of the model I are equivalent or reduce to EDIT problems. The only problem that does not reduce to an edit problem – ALIGN(NEST, NEST, I) → NEST – fully corresponds to the ordered tree alignment, introduced by [89].

The edit model II

In [49], we noticed that the LAPCS problem is a specific case of the common subsequence problem using the edit model II, namely the EDIT(A, B, II) problem, provided that the score system for edit operations is correctly chosen: the cost of a base-deletion or of an arc-altering is 1, the cost of an arc-removing is 2, and substitutions are prohibited, with arbitrary high costs.

Considering this, several cases of the ALIGN hierarchy for the edit model II derive from results published in the LAPCS literature. All known results are summed up in Table 1.10. It [49], we proved that ALIGN(NEST, NEST, II) → NEST can be solved in $\mathcal{O}(n^4)$ using dynamic programming whereas ALIGN(NEST, NEST, II) → CROSS (and consequently, ALIGN(CROSS, NEST, II) → CROSS and ALIGN(CROSS, CROSS, II) → CROSS) is **NP**-complete. These results illustrate the fact that relaxing the constraint on crossing arcs in the common supersequence makes the problem difficult.

$A \times B \rightarrow C$					EDIT	model II
NEST	$\times$	NEST	$\rightarrow$	NEST		$O(n^4)$ – [49]
NEST	$\times$	NEST	$\rightarrow$	CROSS		**NP**-complete – [49]
NEST	$\times$	NEST	$\rightarrow$	UNLIM	$\times$	**NP**-complete – [96]
CROSS	$\times$	NEST	$\rightarrow$	CROSS		**NP**-complete – [49]
CROSS	$\times$	NEST	$\rightarrow$	UNLIM	$\times$	**NP**-complete – [67]
UNLIM	$\times$	NEST	$\rightarrow$	UNLIM	$\times$	
CROSS	$\times$	CROSS	$\rightarrow$	CROSS		**NP**-complete – [49]
CROSS	$\times$	CROSS	$\rightarrow$	UNLIM	$\times$	**NP**-complete – [67]
CROSS	$\times$	UNLIM	$\rightarrow$	UNLIM	$\times$	
UNLIM	$\times$	UNLIM	$\rightarrow$	UNLIM	$\times$	

Table 1.10: ALIGN hierarchy for edit model II. We indicate problems that can be formulated as edit distance problem in the second column. Complexity results are indicated for two arc-annotated sequences (S, P) and (T, Q) s.t. $\max(|S|, |T|) = n$.

Note that the polynomiality of ALIGN(NEST, NEST, II) $\rightarrow$ NEST problem is somehow unexpected since the associate edit problem EDIT(NESTED, NESTED, II) is **NP**-complete. It shows that imposing structural constraints on the type of the common supersequence is an adequate way for achieving lower complexity of untractable problems.

The general edit distance and the edit model III

The edit model III corresponds to the full set of operations. Therefore, several complexity results derive from known results on the *edit distance*. In [49], we prove that ALIGN(NEST, NEST, III) $\rightarrow$ NEST can also be solved in $\mathcal{O}(n^4)$ time.

$A \times B \to C$						EDIT	model III
NEST	$\times$	NEST	$\to$	NEST			$O(n^4)$
NEST	$\times$	NEST	$\to$	CROSS			open
NEST	$\times$	NEST	$\to$	UNLIM		$\times$	**NP**-complete – [33]
CROSS	$\times$	NEST	$\to$	CROSS			
CROSS	$\times$	NEST	$\to$	UNLIM		$\times$	**MaxSNP**-hard – [97]
UNLIM	$\times$	NEST	$\to$	UNLIM		$\times$	
CROSS	$\times$	CROSS	$\to$	CROSS			open
CROSS	$\times$	CROSS	$\to$	UNLIM		$\times$	**MaxSNP**-hard – [97]
CROSS	$\times$	UNLIM	$\to$	UNLIM		$\times$	
UNLIM	$\times$	UNLIM	$\to$	UNLIM		$\times$	

Table 1.11: ALIGN hierarchy for edit model III. We indicate problems that can be formulated as edit distance problem in the second column. Complexity results are indicated for two arc-annotated sequences (S, P) and (T, Q) s.t. $\max(|S|, |T|) = n$.

1.8 Presentation of papers

► Blin, G., Crochemore, M., and Vialette, S. (2011a). *Algorithms in Computational Molecular Biology: Techniques, Approaches and Applications*, chapter Algorithmic Aspects of Arc-Annotated Sequences, pages 113–126. Wiley

This book chapter published in the "Algorithms in Computational Molecular Biology: Techniques, Approaches and Applications" provides an overview of the complexity results regarding the problems involving arc-annotated sequences. The chapter of this thesis is largely based on this contribution.

► Blin, G., Hamel, S., and Vialette, S. (2010c). Comparing RNA structures with biologically relevant operations cannot be done without strong combinatorial restrictions. In Rahman, M. S. and Fujita, S., editors, *4th Workshop on Algorithms and Computation (WALCOM'10)*, volume 5942 of *Lecture Notes in Computer Science*, pages 149–160, Dhaka, Bangladesh. Springer-Verlag

This article presented at the 4^{th} Workshop on Algorithms and Computation (WALCOM'10), Dhaka, Bangladesh focuses on the general edit distance. More precisely, it closes the classical complexity study of the EDIT problem. We prove that using the edit operations allowing to consider either simultaneously or separately letters of a base-pair; unfortunately is done at the cost of computational tractability. [77] have used a strong combinatorial restriction in order to compare two RNA stem-loops with a full set of biologically relevant edit operations; which have allowed

them to design a polynomial-time and space algorithm for comparing general secondary RNA structures. In this paper, we have proved that comparing two RNA structures using a full set of biologically relevant edit operations cannot be done without strong combinatorial restrictions.

► Blin, G., Denise, A., Dulucq, S., Herrbach, C., and Touzet, H. (2010a). Alignment of RNA structures. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 7(2):309–322

► Blin, G. and Touzet, H. (2006). How to Compare Arc-Annotated Sequences: The Alignment Hierarchy. In Crestani, F., Ferragina, P., and Sanderson, M., editors, *13th Symposium on String Processing and Information Retrieval (SPIRE'06)*, volume 4209 of *Lecture Notes in Computer Science*, pages 291–303, Glasgow, UK. Springer

This article published in IEEE/ACM Transactions on Computational Biology and Bioinformatics and presented, in an extended abstract, at the 13^{th} String Processing and Information Retrieval (SPIRE'06), Glasgow, UK describes a theoretical unifying framework to express comparison of RNA structures, which we call ALIGN hierarchy. This framework relies on the definition of common supersequences for arc-annotated sequences, and encompasses main existing models for RNA structure comparison based on trees and arc-annotated sequences with a variety of edit operations. It also gives rise to edit models that have not been studied yet. In this article, we provide a thorough analysis of the alignment hierarchy, including a new polynomial time algorithm and an **NP**-completeness proof. The polynomial time algorithm involves biologically relevant evolutionary operations, such as pairing or unpairing nucleotides. It has been implemented in a software, called gardenia that is available at the web server `http://bioinfo.lifl.fr/RNA/gardenia`.

► Blin, G., Fertin, G., Herry, G., and Vialette, S. (2007d). Comparing RNA structures: towards an intermediate model between the EDIT and the LAPCS problems. In Sagot, M.-F., Walter, W. T., and Maria, E., editors, *1st Brazilian Symposium on Bioinformatics (BSB), Angra dos Reis, Brazil*, volume 4643 of *Lecture Notes in Bioinformatics*, pages 101–112. Springer

This article presented at the 1^{st} Brazilian Symposium on Bioinformatics (BSB), Angra dos Reis, Brazil introduces a new and general intermediate model for comparing RNA structures: the MAXIMUM ARC-PRESERVING COMMON SUBSEQUENCE problem (or MAPCS). This new model lies between two well-known problems – namely the Longest Arc-Preserving Common Subsequence (LAPCS) and the EDIT distance. After showing the relationship with other paradigms, we investigate the computational complexity landscape of MAPCS, depending on the RNA structure complexity.

► Blin, G., Fertin, G., Rusu, I., and Sinoquet, C. (2007e). Extending the Hardness of RNA Secondary Structure Comparison. In Bo, C., Mike, P., and Guochuan, Z., editors, *1st intErnational Symposium on Combinatorics, Algorithms, Probabilistic and Experimental methodologies (ESCAPE'07)*, volume 4614 of *LNCS*, pages 140–151, Hangzhou, China, Chine. Springer-Verlag

This article presented at the 1^{st} intErnational Symposium on Combinatorics, Algorithms, Probabilistic and Experimental methodologies (ESCAPE'07), Hangzhou, China considers the EDIT distance. In this contribution, we prove that EDIT(NESTED, NESTED) is **NP**-complete for a

large class of instances, not overlapping with the ones used in the proof for LAPCS, and which represent more biologically relevant cost systems; hence, giving a more precise categorization of the computational complexity of the EDIT problem.

2

Genomes Comparison: Permutations and Sequences

Contents

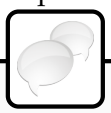
2.1 Introduction

Comparative genomics is an active field of bioinformatics. One of the problems arising in this domain consists in comparing two or more species. The main ways of comparing them are either by seeking for *gene clusters* or by computing *genomic distances* between their genomes. A gene cluster refers to a set of genes appearing, in spatial proximity along the chromosome, in at least two genomes. Genomes evolved from a common ancestor tend to share some gene clusters. Therefore, they may be used for reconstructing recent evolutionary history and inferring putative functional assignments for genes.

There are numerous ways of mathematical formalizations of gene clusters. Among others, one can mention *common substrings* (which require a full conservation), *common intervals* [124; 115; 62; 9; 25] (genes must occur consecutively, regardless of their order), *conserved intervals* [13; 7] (common intervals, framed by the same two genes), *gene teams* [11; 81; 131] (genes in a cluster must not be interrupted by long stretches of genes not belonging to the cluster), and *approximate common intervals* [108; 51; 130] (common intervals that may contain few genes from outside the cluster).

Computing a (dis)similarity measure that approximates the true evolutionary distance between the genomes mainly arise in phylogeny reconstruction. Most of the mathematical models developed so far to compute such (dis)similarity measures are based on the assumption that genomes are represented as permutations (allowing a one-to-one correspondence between genes of different genomes). However, aside some particular cases such as mitochondrial genomes [113], due to

genome evolution process, including – among others – fundamental evolutionary events such as gene duplication and loss [103], duplicated genes are very common in genomes.



Preliminary to most comparative genomics studies is the annotation of chromosomes as ordered sequences of genes. Unfortunately, different genetic mapping techniques usually give rise to different maps with unequal gene content, and often containing sets of unordered neighboring genes. Only partial orders can thus be obtained from combining such maps. Leading to a directed acyclic graph representation rather than a permutation. However, once a total order O is known for a given genome, it can be used as a reference to order genes of a closely related species characterized by a partial order P. Interested readers in such aspects may refer to [16], where we investigated the problem of finding a linearization of P that is as close as possible to O in terms of the breakpoint and common interval distances that will not be presented in this manuscript.

As a result, when using real data, one has to deal with the fact that genomes are modeled by sequences of integers – where the same integer (*i.e.* gene) may occur more than once (a more realistic model but with higher complexity). Such genes that appear at several occurrences are said to belong to non-trivial gene families. In both models, there may exist (or not) genes that are not shared between two genomes (often called *gaps*). Moreover, when modeling genomes for gene order analysis, one may consider either *two* or *multiple* genomes, seeking for *exact* or *approximate* occurrences, finding *all* or just non-extensible (*i.e. maximal*) occurrences. Therefore, computing (dis)similarity measures between genomes relies on a two steps permutation based approach.

The first step consists in transforming the two sequences into a single permutation P by establishing a one-to-one correspondence between pairs of genes having the same label (and then, by resorting to some renaming procedure, we can always assume that one of the two permutations is the identity permutation). In the second step, a permutation-based (dis)similarity measure is computed from the permutation P. The main line of research following this approach seeks for the permutation P that optimizes the (dis)similarity measure for a given matching model. The classical criterion retained to define the optimal (dis)similarity measure is the parsimony criterion: one tries to compute the permutation P that induces the maximal (resp. minimal) similarity (resp. dissimilarity) measure.

There are two main approaches for computing a one-to-one correspondence between two integer sequences. In the *exemplar* model, introduced by [110], for every non-trivial gene family, all but one copy in each genome are deleted. The pair of genes that is conserved for each family is called a pair of ancestral homologs, as the goal of the exemplar method is to find the pair of genes which best reflects the original position of the ancestral gene in the common ancestor genome. The *matching* model is more general as it allows to conserve more than one copy of a gene family and seeks for a maximal one-to-one correspondence between these copies [19]. Several distances have been considered under the exemplar and matching models, that are either based on minimizing the number of evolutionary events that allow to transform a genome into the other, for events

like reversals[110; 56; 59; 120; 101; 61], reversals and insertions and deletions [99; 121], reversals and translocations [72], or on maximizing a similarity measure based on conserved structure in permutations like the number of adjacencies (which is equivalent to minimizing the number of breakpoints) [110; 56; 102; 101; 61; 132] or the number of conserved intervals [40; 54; 60; 7]. As far as we know, none of the above problems has been shown to be solvable in polynomial time as soon as duplicates are present in genomes.

Remark that duplication are also of importance when trying to compute evolutionary history of species. Indeed, the evolutionary history of the genomes of eukaryotes is the result of a series of evolutionary events, called speciations, that produce new species starting from a common ancestor. This evolutionary history has been deeply studied in Computational Biology, and is usually represented using a special type of phylogenetic tree called species tree [70]. A species tree is a rooted binary tree whose leaves are uniquely labelled by a set representing the extant species, where the common ancestor of the contemporary species is associated with the root of the tree. The internal nodes represent hypothetical ancestral species (and the associated speciations). Speciations are not the only events that influence the evolution. Indeed, among other, gene duplications, although not leading to new species, are fundamental in evolution. Remember that, gene duplication can be described as the genomic event that causes a gene inside a genome to be copied, resulting in two copies of the same gene that can evolve independently. Genes of extant species are called homologous if they evolved from a common ancestor, through speciations and duplications events [71]. Evolution of homologous genes, with regards to the extant species, is usually represented using another special type of phylogenetic tree called gene tree. Due to complex evolutionary processes such as gene duplication and loss, comparable gene and species trees very often present incompatibilities. A challenging problem is then to reconcile the gene and species trees with hypothetical gene duplications – referred as the MINIMUM DUPLICATION PROBLEM. Interested reader may refer to [17], where we investigated the inapproximability of the problem.

This chapter is devoted to presenting algorithmic results for finding gene clusters and computing (dis)similarity measures and organized as follows. Section 2.2 is devoted to problems related to Gene Clusters detection whereas we consider in Section 2.3 genomic (dis)similarity measures computation.

2.2 Detecting Gene Clusters

The genetic blueprint of an organism is encoded in a set of DNA sequences, known as chromosomes. During evolution, some subsequences of a chromosome diverged while others were conserved among different organisms. Many of these conserved subsequences correspond to functional

elements – referred to as *genes*, which are of paramount importance in understanding evolution. Therefore, in many studies, a chromosome is represented as a sequence of genes and evolution is described as a series of discrete events, such as gene insertion, loss, duplication and inversion. Two genes with highly similar sequences, typically arising via speciation or duplication, are considered as belonging to the same gene family. In this chapter, a gene family and its constituent genes are assigned the same label. One of the most important goals in comparative genomics is to identify a set of genes that are in proximate locations on multiple chromosomes and their actual chromosomal occurrences. Indeed, preservation of gene co-locality tends to indicate that the corresponding genes either form a functional unit (*e.g.*, operons) or result from speciation or duplication events [84]. In the literature, the former is termed "*gene cluster*", whereas the latter is known as "*synteny*" [129]. Both were extensively studied during the past decade, and numerous models and algorithms were proposed to define and identify them. Most gene cluster models are formally defined [10] while many synteny detection methods are *ad hoc* and lack formal definitions. In this chapter, we will focus only on formally defined models and present the corresponding results from an algorithmic point of view.

2.2.1 Gene Proximity: Properties and Models

Modeling gene proximity based on biological intuition is known to be difficult. Nevertheless, some key properties have been raised by [84]. Let us present briefly these properties.

Key Properties of Gene Proximity

The first crucial property consists in *evidence of any gene of interest as being ancestral*. This property is usually related to observing a minimum number of β occurrences of such a gene, thereby reducing the possibility of misinterpreting what is in fact a chance occurrence.

Based on the fact that genes of interest appear, with relative proximity, in different chromosomes, most of the models consider chromosomal regions, usually referred as *segments* or *intervals*, as being of interest. Naturally, such segments are subjected to constraints in order to confirm their common origin. First of all, each *contributes sufficiently* to the ancestral gene set. More formally, it means that such a segment has to contain a minimum number of ϵ_m different ancestral genes. Then, considering evolutionary events that may have occurred, those segments may not necessarily contain all the ancestral genes (*i.e.*, gene losses). Meanwhile, they may contain genes not belonging to the ancestral gene set (*i.e.*, gene insertions).

For the segment to be relevant, some constraints on gene insertions and losses have to be imposed, which are referred to as *local* and *global ancestral gene densities*. The local density is captured by a maximum number of interleaving genes between two consecutive ancestral genes in a segment (usually referred as α). On the other hand, global density is captured by the maximum number ϵ_l of gene losses in the segment and the maximum overall number ϵ_t of gene losses of all segments of interest. One can easily conceive that ϵ_l and ϵ_t play different roles: while ϵ_l controls locally in a segment the preservation of a maximum number of ancestral genes, constraining only ϵ_t may allow for a long unconserved region to occur within some segment of interest.

Existing Models

Consider k chromosomes, each given as a permutation over a given gene set $\mathcal{G}$. A CONSERVED SEGMENT [100] consists in a set of genes that occur consecutively in the same order on every input chromosome. Once the constraint of the preserved ordering is removed, it leads to the COMMON INTERVAL (CI) model definition [125]. If the unordered pair of the first and the last genes of a CI is the same on each chromosome, this CI is moreover called *conserved* [13]. Furthermore, if we relax the constraint that genes in a CI have to be consecutive in each chromosomal occurrence – namely, two genes belonging to a CI can be interleaved by a bounded number of genes not belonging to it – the definition of GENE-TEAMS (GT) (also referred as MAX-GAP) model [11] follows. The GT model is of higher biological relevance compared to the CI model since it further captures gene insertions, *i.e.*, genes not belonging to the CI.

For example, given G = 1 9 3 4 5 6 7 8 2 and H = 1 2 7 4 5 6 8 3, $\{4,5,6\}$ is a CONSERVED SEGMENT, $\{4,5,6,7,8\}$ is a COMMON INTERVAL, $\{2,3,4,5,6,7,8\}$ is a CONSERVED INTERVAL, $\{1,2,3,4,5,6,7,8\}$ is a GENE-TEAM with one gap.

So far, we assumed chromosomes as gene permutations, which is rarely the case in practical application. To elevate the biological accuracy, chromosomes are represented as strings over gene set $\mathcal{G}$ such that multiple occurrences of genes, arising via duplication events, can occur on the same chromosome. The aforementioned model definitions naturally apply to strings, but the number of gene sets complying with the model may increase significantly.

More recently, APPROXIMATE COMMON INTERVAL (ACI) models were introduced [4; 108; 51]. Unlike previous models, not all genes of interest have to be present in every chromosomal occurrence. MEDIAN GENE CLUSTER (MGC) model [51] is the most recent formulation of ACI in which the problem is to identify in chromosomes (represented as strings) $S_1, S_2, \ldots S_k$, the gene set $\mathcal{A}$ (of interest) and its chromosomal occurrences $S'_1, S'_2, \ldots S'_k$ satisfying $\sum_{i=1}^{k} (|\mathcal{A} \setminus \mathcal{CS}(S'_i)| + |\mathcal{CS}(S'_i) \setminus \mathcal{A}|) \leq \delta$ and $\sum_{i=1}^{k} (|\mathcal{A} \setminus \mathcal{CS}(S'_i)| + |\mathcal{CS}(S'_i) \setminus \mathcal{A}|) \leq \sum_{i=1}^{k} (|\mathcal{A}' \setminus \mathcal{CS}(S'_i)| + |\mathcal{CS}(S'_i) \setminus \mathcal{A}'|)$, for all $\mathcal{A}' \subseteq \mathcal{G}$. In this formulation, S'_i is a substring of S_i; $\mathcal{CS}(S')$ denotes the character set (or gene set) of S'; and δ is the maximum overall content difference allowed between $\mathcal{A}$ and the S'_is. In addition, $|\mathcal{A}|$ has to be large enough to be biologically meaningful. Note that according to this definition, any character of $\mathcal{A}$ has to belong to at least $\frac{k}{2}$ substrings.

Recently, with X. Yang, F. Sikora, S. Hamel, and S. Aluru, we [130] proposed a new attempt to formalize the biological intuition of gene proximity modeling in the notion of MULTI-RELATED-SEGMENTS (MRS). Similar to other models, a MRS can be defined as consisting of a set of segments of interest, each evolved from an ancestral segment with gene set $\mathcal{A}$ by gene insertion, loss, duplication, and inversion events. Formally, a MRS is defined as follows. To ensure *evidence of being ancestral genes*, any gene belonging to $\mathcal{A}$ has to occur in at least β (≥ 2) segments. Each segment of interest has to contain at least ϵ_m different ancestral genes and be maximal (*i.e.*, not extendable by including surrounding genes) – thus, imposing a constraint on the *minimum contribution to* $\mathcal{A}$. As previously done in the GT model, the *local ancestral gene density* is obtained by an upper bound α controlling the number of non-ancestral genes between any two consecutive ancestral ones in each segment. To capture *global ancestral gene density*, we require each segment to induce no more than ϵ_l gene losses and the total number of gene losses of all segments to be lower than ϵ_t. Then, given a set of chromosomes and parameters $\alpha, \beta, \epsilon_m, \epsilon_l$ and ϵ_t, the general problem is to identify all MRS.

Compared to existing models, the MRS definition has the following biological advantages. First of all, it captures previous models. MRS corresponds to a CI when $\beta = k$, $\epsilon_m = |\mathcal{A}|$ and $\alpha = 0$, and to a GT when $\alpha \geq 0$. Compared to these two models, MRS further captures gene loss events. Note that this aspect was already considered in the MGC model [51]. Nevertheless, there are several major differences. Firstly, MRS captures the same origin of more than two segments in the absence of strong pairwise similarity information, such as differential gene loss [119] and uber-operon [58] – which is not the case for MGC due to the requirement that segments pairwisely share some common genes. Moreover, the minimum evidence of a gene being ancestral is more flexible in MRS by requiring β occurrences of any ancestral gene – which has to be at least $\frac{k}{2}$ in MGC. Finally, the local ancestral gene density is not required in MGC – which is, as explained in [84], crucial.

2.2.2 Know results

Common Intervals

[125] first introduced the notion of common interval in order to capture, when comparing genomes, that a set of genes may have been rearranged while remaining relatively close one of each other. [125] designed an algorithm computing the set of common intervals of a permutation P (*w*.r.t., the identity permutation) in $\mathcal{O}(n + N)$ time; n and N being respectively the length of P and the number of common intervals. In other words, the complexity of their algorithm relies on the size of the output. Thus, since N can be of size $\mathcal{O}(n^2)$, [125] algorithm has a $\mathcal{O}(n^2)$ time complexity.

Later on, given k permutations of n elements, [82] proposed an improvement of [125] approach by a non-trivial extension, yielding an optimal $\mathcal{O}(kn + K)$ time and $\mathcal{O}(n)$ space algorithm, where K is the number of common intervals. The approach relies on restricting the set of all common intervals C to a smaller subset of *irreducible intervals* from which C canj be easily reconstructed. To do so, [82] algorithms rely on a complex data structure related to PQ-trees [94].

An alternative efficient algorithm was proposed by [9]. Indeed, [9] proposed a theoretical framework for computing common intervals based on a linear space basis. Of importance here is the technique proposed in [9] to generate the PQ-tree corresponding to a linear space basis for computing all the common intervals of K permutations. Generating this basis can be done in $\mathcal{O}(n)$ time for two permutations of size n. Then one can, by a browsing of the tree, generate all the common intervals in $\mathcal{O}(n + N)$ time, where N is the size of the output.

Following [84] line of reasoning, together with D. Faye and J. Stoye, we [25] considered another model – namely the NESTED COMMON INTERVALS (NCI). In this model, an additional constraint – called the *nestedness* – is added to the cluster definition. Roughly speaking, a common interval C is called a nested common interval of two permutations if either it is of length two or it contains a nested common interval of size $|C| - 1$. [84] argued that, depending on the dataset, if the nestedness assumption is not excluding clusters from the data, then it can strengthen the significance of detected clusters since it reduces the probability of observing them by chance. For permutations, we [25] gave several algorithms whose running time depends on the size of the actual output rather than the output in the worst case. Indeed, we first provided a straightforward cubic time algorithm for finding all nested common intervals; it was reduced to a quadratic time algorithm for irredundant output. A third algorithm shows that finding only the maximal nested common intervals can be done in linear time. Finally, we proved that finding approximate nested common

intervals is fixed-parameter tractable. For sequences, we provided solutions for different variants of the problem, depending on the treatment one wants to apply to duplicated genes (the uniqueness, the free-inclusion, or the bijection models). This includes a polynomial-time algorithm for a variant implying a matching of the genes in the cluster, a setting that for other problems often leads to hardness. Recently, [128] further investigated the problem of finding all nested common intervals of two general sequences. For the uniqueness and the bijection models, [128] gave $\mathcal{O}(n + N_{out})$-time algorithms, where N_{out} denotes the size of the output. For the free-inclusion model, [128] gave an $\mathcal{O}(n^{(1+e)} + N_{out})$-time algorithm, where $e > 0$ is an arbitrarily small constant.

Conserved Intervals

[13] tackled the conserved intervals detection and designed linear algorithms adapted from [12] that outputs the irreducible conserved intervals of two permutations P and Q in $\mathcal{O}(n)$ time, and the irreducible conserved intervals of a set of k permutations in $\mathcal{O}(kn)$ time.

Gene Teams

[11] relaxed the "consecutive" constraint by introducing gene teams – allowing genes in a cluster to be separated by gaps that do not exceed a fixed threshold – and presented an $\mathcal{O}(kn \log^2 n)$ time algorithm for finding all gene teams of k permutations of n elements. Notice [105] proved the problem to be exponential for strings.

Approximate Common Intervals

[4] proposed a $\mathcal{O}(kn^3 + occ)$ time algorithm for k strings of n elements, where occ is the output size. However, it is possible to construct a counter example for which their graph-based algorithm does not detect the complete solution, as pointed out by [85].

[108] contribute to the discussion about the concept of approximate conserved gene clusters by presenting a class of definitions that (1) can be written as integer linear programs (ILPs) and (2) allow several variations that include existing definitions such as common intervals and max-gap clusters or gene teams. This ILP formulation provides unprecedented generality and is competitive in practice for those cases where efficient algorithms are known.

[51] introduced a new cluster concept, named MEDIAN GENE CLUSTERS, that constrains only the sum of errors that may occur in the approximate occurrences of a gene cluster and designed an algorithm leading to $\mathcal{O}(n^2(1+\delta)^2)$ time and $\mathcal{O}(n^2)$ space complexities with $\delta \ll n$.

With X. Yang, F. Sikora, S. Hamel, R. Rizzi and S. Aluru, we [130] considered from an algorithmic point of view the problem of MRS inference by applying various restrictions on the model definition. We showed that modeling gene losses turns both search scenarios, where an ancestral gene set is given, and the general case, into computationally hard problems for the MRS model. The former is shown to be fixed-parameter tractable and the latter to be **APX**-hard.

2.3 Computing (dis)similarity measures

In this section, we define the five similarity measures we are interested in. As mentioned in the introduction, most considered measures are defined for duplication-free genomes of equal gene contents only, and hence one has first to disambiguate the data by inferring homologs, *i.e.*, a non-ambiguous mapping between the genes of the two genomes. Note that differences in gene contents are often also took into account in the computation of (dis)similarity measures as an extra penalty.

Let us introduce some notations. Considering *genomes* as sequences of unsigned integers, let G be a genome of size n. As mentioned above, a *gene family* is any integer that occurs in G, regardless to its number of occurrences. A *gene* is an occurrence of a gene family in G, and we denote by $G[i]$ the gene that occurs at position i in G. Let $\mathrm{occ}(G, g)$ denote the maximum number of occurrences of a gene g in genome G, and let $\mathrm{occ}(G)$ be the maximum of $\mathrm{occ}(G, g)$ over all genes g in G. The genome G is said to be *duplication-free* if $\mathrm{occ}(G) = 1$. Now, let G_1 and G_2 be two genomes. A *matching* $\mathcal{M}$ between G_1 and G_2 is a set of pairwise disjoint pairs $\mathcal{M} = \{(i_1, j_1), (i_2, j_2), \ldots, (i_k, j_k)\}$ such that $G_1[i_\ell] = G_2[j_\ell]$ for all $1 \leq \ell \leq k$. Suppose that G is duplication-free ; let $1 \leq i < j \leq n$ such that. $a = G[i]$ and $b = G[j]$. The *distance* between a and b in G, written $\mathrm{Dist}(G, a, b)$, is defined by $\mathrm{Dist}(G, a, b) = |j - i|$.

Given two genomes containing duplications, a first step is thus to establish a non-ambiguous mapping between the genes of the two genomes. In the *exemplar* model, for all gene families, all but one occurrence in each genome are deleted. In other words, we are looking for a matching $\mathcal{M} = \{(i_1, j_1), (i_2, j_2), \ldots, (i_k, j_k)\}$ between G_1 and G_2 such that (i) $G_1[i_\ell] \neq G_1[i_{\ell'}]$ for all $1 \leq \ell < \ell' \leq k$ and (ii) each gene family occurs in one pair of $\mathcal{M}$. In the *matching* model, the goal is to map as many genes as possible, *i.e.*, find a maximum matching between G_1 and G_2. The rationale of this preliminary step is that we may now assume that the two genomes are duplication-free and of equal gene content (in terms of alphabet). Indeed, suppose the first step results in the matching $\mathcal{M}$, we thus modify the two genomes G_1 and G_2 as follows:

1. we delete all genes in G_1 and G_2 that are not part of the matching $\mathcal{M}$, and

2. we rename the genes of G_1 and G_2 according to the index of the associated pair in $\mathcal{M}$.

Observe that the resulting genomes are both of size $|\mathcal{M}|$. According to the above (for both the exemplar and the matching models), if a gene family occurs in one genome but not in the other then all occurrences of this gene family will be deleted in the end. Therefore, we may thus assume in the sequel that any gene family of G_1 is a gene family of G_2, and conversely. An illustration is given in Figure 2.1

In order for the corresponding mapping to be relevant, one has to rely on a (dis)similarity measure. The problem then corresponds to, given two genomes and a measure definition, finding a mapping that will induce the optimal corresponding measure. For (dis)similarity measures, one can mainly distinguish two main classes of measures: the ones based on the seek of similar regions (usually represented as intervals) and the ones based on (un)conserved adjacencies.

We now turn to define the five similarity measures we are interested in. As mentioned before, we may assume now that the two genomes are duplication-free, *i.e.*, both G_1 and G_2 are permutations

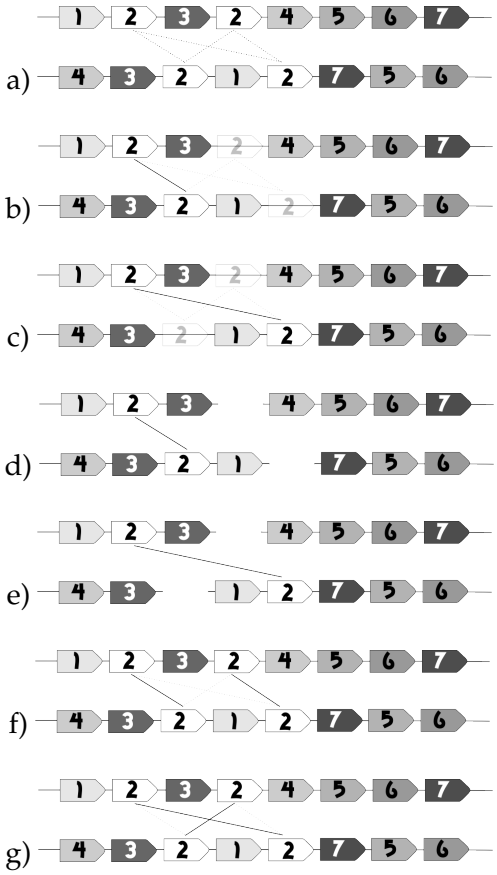


Figure 2.1: Given two genomes (a), in the exemplar model (b,c,d,e), in each genome, all but one occurrence of any gene are deleted. In the matching model (f,g), a set of pairwise disjoint pairs of genes are matched.

of size n. Moreover, for convenience, by first resorting to an easy renaming procedure we can always assume that one of the two genomes, say G_1, is the identity permutation, *i.e.*, $G_1 = 1\ 2\ \ldots\ n$.

2.3.1 Seeking for similar regions

The corresponding measures refer to a specific gene cluster model (as presented in Section 2.2) and mainly count their occurrences. Despite the fact that any such model may be used, mainly common and conserved intervals were studied. With C. Chauve, G. Fertin, R. Rizzi ans S. Vialette, we [21] investigated the algorithmic complexity of computing the number of common intervals between two genomes, in both the exemplar and matching models and proved that the problem was **NP**-complete for both models and even for restricted instances (namely $\mathrm{occ}(G_1) = 1$ and $\mathrm{occ}(G_2) = 2$).

For the matching model, we [21] considered, instances for which the constraints do not rely on the maximum number of duplicates per family, but on the number of families that contain

duplicates. With this restriction, we proved the **NP**-completeness of the problem, even when $f(G_1) = f(G_2) = 1$, where $f(G)$ denotes the number of different families of genes that contain duplicates in G. This proof was directly derived from the proof of [40], in which we studied conserved intervals. Indeed, any conserved interval is by definition a common interval, though the converse is not true in general. However, the construction given in [40] has the property that any common interval is in fact also a conserved interval; therefore proving that the reduction provided holds for common intervals.

2.3.2 Seeking for (un)conserved adjacencies

For the (un)conserved adjacencies, one can mention mainly two types of measures: breakpoints and adjacency disruption.

Number of breakpoints

Given two genomes G_1 and G_2, built over the same alphabet, a breakpoint in G_1 corresponds to a pair of consecutive genes (*e.g.* $(G_1[i], G_1[i+1])$) that are not consecutive in G_2. For instance, if $G_1 = 1\,2\,3\,4\,5$ and $G_2 = 1\,4\,3\,5\,2$ then $\{(1,2),(2,3),(4,5)\}$ are breakpoints. The EXEMPLAR BREAKPOINT DISTANCE problem asks whether it is possible to establish an exemplar matching of G_1 and G_2, such that the number of breakpoints between the resulting genomes is at most k.

[56] showed that EXEMPLAR BREAKPOINT DISTANCE is **NP**-complete, even when one of the genomes is trivial, and the other one has genes that appear at most twice in each genome. For (in)approximability results, [7] proved that EXEMPLAR BREAKPOINT DISTANCE is **APX**-hard under the same assumptions. [61] provided a logarithmic approximation ratio for the particular case in which one of the genomes is an s-span genome, with $s = O(\log m)$, $m = |\Sigma|$. A genome G is called an s-span genome if all the genes from the same gene family are within distance at most s in G.

It should also be noted that [102] designed a divide-and-conquer heuristic method in order to compute the exemplarization while [6] proposed an exact method based on transforming the problem into a 0-1 linear programming problem. [61] also showed that there exists no approximation algorithm for EXEMPLAR BREAKPOINT DISTANCE, even when both genomes have genes that appear at most three times. With G. Fertin, F. Sikora and S. Vialette, we [34] tighten this result by proving that no approximation factor can be derived for EXEMPLAR BREAKPOINT DISTANCE, even for genomes in which each gene occurs at most twice; that is the simpliest case of non-trivial genomes.

To do so, we [34] proved that a particular subproblem of EXEMPLAR BREAKPOINT DISTANCE – called the ZERO EXEMPLAR BREAKPOINT DISTANCE problem (ZEBD for short) – is **NP**-complete. This decision problem asks whether there exists an exemplar matching of two genomes, such that the breakpoint distance between the resulting genomes is equal to zero. For sake of readability, for any $1 \leq p \leq q$, we will write ZEBD(p, q) for the ZEBD problem in which $occ(G_1) = p$ and $occ(G_2) = q$. It is easy to see that ZEBD$(1, q)$ can be solved in linear time, for any $q \geq 1$. [61] showed that ZEBD$(3, 3)$ is **NP**-complete. [7] also showed that ZEBD$(2, q)$ is **NP**-complete, but with a value of q unbounded due to their reduction. We [34] proved that ZEBD$(2, 2)$ (and thus, ZEBD$(2, q)$ for any $q \geq 2$) is **NP**-complete.

Adjacency Disruption Number

Suppose that G is duplication-free ; let $1 \leq i < j \leq n$, $a = G[i]$ and $b = G[j]$. The Adjacency Disruption Number between a and b in G – introduced by [111] – written $\mathtt{Dist}(G, a, b)$, is defined by $\mathtt{Dist}(G, a, b) = |j - i|$. [111] defined two related dissimilarity measures; namely MAD and SAD.

The *Maximum Adjacency Disruption Number* (MAD) number between G_1 and G_2, denoted $\mathrm{MAD}(G_1, G_2)$, is defined by

$$\mathrm{MAD}(G_1, G_2) = \max\{\mathcal{M}_1, \mathcal{M}_2\},$$

where $\mathcal{M}_1 = \max\{\mathrm{Dist}(G_2, G_1[i], G_1[i+1]) : 1 \leq i \leq n-1\}$ and $\mathcal{M}_2 = \max\{\mathrm{Dist}(G_1, G_2[i], G_2[i+1]) : 1 \leq i \leq n-1\}$.

It roughly corresponds to the maximum distance of a pair of genes g and h in G_1, G_2, for any g and h such that g and h are adjacent in one of $\{G_1, G_2\}$. The rationale of this double maximization measure lies in the fact that, in general, $\mathcal{M}_1 \neq \mathcal{M}_2$. For instance, if $G_1 = 1\,2\,3\,4\,5$ and $G_2 = 1\,4\,3\,5\,2$ then $\mathcal{M}_1 = 4$ and $\mathcal{M}_2 = 3$, and hence $\mathrm{MAD}(G_1, G_2) = \max\{4, 3\} = 4$.

The *Summed Adjacency Disruption Number* (SAD) number – that can be seen as a global variant of the MAD number – between G_1 and G_2, denoted $\mathrm{SAD}(G_1, G_2)$, is defined by

$$\mathrm{SAD}(G_1, G_2) = \sum_{1 \leq i \leq n-1} \mathrm{Dist}(G_2, G_1[i], G_1[i+1]) + \sum_{1 \leq i \leq n-1} \mathrm{Dist}(G_1, G_2[i], G_2[i+1])$$

Going back to our example $G_1 = 1\,2\,3\,4\,5$ and $G_2 = 1\,4\,3\,5\,2$, one obtains $\mathrm{SAD}(G_1, G_2) = (4 + 2 + 1 + 2) + (3 + 1 + 2 + 3) = 18$.

Recall that $\mathrm{occ}(G)$ denotes the maximum of $\mathrm{occ}(G, g)$ over all genes g in G, where $\mathrm{occ}(G, g)$ denotes the maximum number of occurrences of a gene g in genome G. We also recall that $f(G)$ denotes the number of different families of genes that contain several occurrences in genome G.

With C. Chauve, G. Fertin, R. Rizzi ans S. Vialette, we [21] proved that both for the exemplar and matching models computing SAD and MAD numbers are **NP**-complete and **APX**-hard problems. More precisely, we proved that the **NP**-hardness and **APX**-hardness of MAD (resp. SAD) hold even when $\mathrm{occ}(G_1) = 1$ and $\mathrm{occ}(G_2) \leq 9$ (resp. $\mathrm{occ}(G_1) = 1$).

The main conclusion that we can draw is that, as soon as $occ(G_1) = 1$ and $occ(G_2) = 2$, the computation of the previously mentioned measures becomes **NP**-complete, for both the exemplar and matching models. In that sense, we are able to draw the exact border between polynomial problems ($occ(G_1) = occ(G_2) = 1$) and **NP**-complete problems ($occ(G_1) = 1$ and $occ(G_2) = 2$). Another interesting parameter to consider for the complexity of those problems is $f(G)$, the number of families of genes that are duplicated in genome G. Concerning this parameter, only a few results are known (breakpoints, conserved and common intervals, in the matching model only). Concerning the approximability of the problems, it turns out that even when $occ(G_1) = 1$, the computation of the previously mentioned measures lead to **APX**-hard problems. More precisely, for breakpoints, conserved or common intervals, we know that the problem is **APX**-hard even when $occ(G_1) = 1$ and $occ(G_2) = 2$, while the value of $occ(G_2)$ is either unbounded or bounded by constant 9 for respectively SAD and MAD.

2.4 Presentation of papers

► Blin, G., Bonizzoni, P., Dondi, R., Rizzi, R., and Sikora, F. (2012a). Complexity Insights of the Minimum Duplication Problem. In Bieliková, M., Friedrich, G., Gottlob, G., Katzenbeisser, S., and Turán, G., editors, *38th International Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM 2012)*, volume 7147 of *LNCS*, pages 153–164, Špindlerův Mlýn, Tchèque, République. Springer-Verlag

This article presented at the 38th International Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM 2012) in Špindlerův Mlýn, Czech Republic investigates a well-known problem in phylogenetics and comparative genomics: the so-called MINIMUM DUPLICATION problem. Given a set of gene trees, the MINIMUM DUPLICATION problem asks for a species tree that induces the minimum number of gene duplications in the input gene trees. More recently, a variant of the MINIMUM DUPLICATION problem, called MINIMUM DUPLICATION BIPARTITE, has been introduced by [104], where the goal is to find all pre-duplications, that is duplications that precede, in the evolution, the first speciation with respect to a species tree. In this paper, we investigate the complexity of both MINIMUM DUPLICATION and MINIMUM DUPLICATION BIPARTITE problems. First of all, we prove that the MINIMUM DUPLICATION problem is **APX**-hard, even when the input consists of five uniquely leaf-labelled gene trees (progressing on the complexity of the problem). Then, we show that the MINIMUM DUPLICATION BIPARTITE problem can be solved efficiently by a randomized algorithm when the input gene trees have bounded depth.

► Blin, G., Bonizzoni, P., Dondi, R., and Sikora, F. (2012b). On the Parameterized Complexity of the

Repetition Free Longest Common Subsequence Problem. *Information Processing Letters*

This article published in Information Processing Letters considers the Repetition Free Longest Common Subsequence problem (RFLCS). RFLCS is a variant of the LCS problem that asks for a longest common subsequence problem of two input strings with no repetition of symbols. In this paper, we investigate the parameterized complexity of RFLCS, by first, proving that the problem does not admit a polynomial kernel and, giving an FPT algorithm for the RFLCS problem, improving the time complexity of the best known FPT algorithm.

► Blin, G., Rizzi, R., and Vialette, S. (2010d). A faster algorithm for finding minimum Tucker submatrices. In *6th Computability in Europe (CiE'10)*, volume 6158 of *Lecture Notes in Computer Science*, pages 69–77, Portugal. Springer
► Blin, G., Vialette, S., and Rizzi, R. (2012d). A faster algorithm for finding minimum Tucker submatrices. *Theory of Computing Systems*, page 10 pp
► Blin, G., Rizzi, R., and Vialette, S. (2011e). A polynomial-time algorithm for finding minimal conflicting sets. In Kulikov, A. and Vereshchagin, N., editors, *Proc. 6th International Computer Science Symposium in Russia (CSR)*, volume 6651 of *Lecture Notes in Computer Science*, pages 373–384. Springer

These articles presented respectively at the 6^{th} Computability in Europe (CiE'10), Portugal and at the 6^{th} International Computer Science Symposium in Russia (CSR'11), St Petersbourg, Russia study the so-called C1P property of binary matrices. A binary matrix has the Consecutive Ones Property (C1P) if there exists a permutation of its columns (*i.e.* a sequence of column swappings) such that in the resulting matrix the 1s are consecutive in every row. Algorithmic issues of the C1P are central in computational molecular biology, in particular for physical mapping and ancestral genome reconstruction. In 1972, Tucker gave a characterization of matrices that have the C1P by a set of forbidden submatrices, and a substantial amount of research has been devoted to the problem of efficiently finding such a minimum size forbidden submatrix. The former paper presents a new $O(\delta^3 m^2(m\delta + n^3))$ time algorithm for this particular task for a $m \times n$ binary matrix with at most δ 1-entries per row, thereby improving the algorithm of [63]. The latter paper further investigate the potential of C1P in ancestral genome reconstruction by studying the problem of finding MCS. A Minimal Conflicting Set (MCS) of rows is a set of rows R that does not have the C1P, but such that any proper subset of R has the C1P. [57] gave an $O(\delta^2 m^{max(4,\delta+1)}(n + m + e))$ time algorithm to decide if a row of a $m \times n$ binary matrix with at most δ 1s per row belongs to at least one MCS of rows. We present the first polynomial-time algorithm to decide if a row of a $m \times n$ binary matrix belongs to at least one MCS of rows.

► Blin, G., Rizzi, R., Sikora, F., and Vialette, S. (2011c). Minimum Mosaic Inference of a Set of Recombinants. *International Journal of Foundations of Computer Science (IJFCS)*. To appear
► Blin, G., Rizzi, R., Sikora, F., and Vialette, S. (2011d). Minimum Mosaic Inference of a Set of Recombinants. In Alex, P. and Taso, V., editors, *17th Computing: the Australasian Theory Symposium (CATS'11)*, volume 119 of *CRPIT*, pages 23–30, Perth, Australie. ACS

This article presented at the 17^{th} Computing: the Australasian Theory Symposium (CATS'11),

Perth, Australia and extended for a journal version in International Journal of Foundations of Computer Science investigate the central problem of finding recombination events. It is commonly assumed that a present population is a descendent of a small number of specific sequences called founders. Due to recombination, a present sequence (called a recombinant) is thus composed of blocks from the founders. A major question related to founder sequences is the so-called MINIMUM MOSAIC problem: using the natural parsimony criterion for the number of recombinations, find the "best" founders. In this article, we prove that the MINIMUM MOSAIC problem given haplotype recombinants with no missing values is hard for an unbounded number of founders and propose some exact exponential-time algorithms for the problem. Notice that, Rastas et al. proved that the MINIMUM MOSAIC problem is hard using a somewhat unrealistic mutation cost function (details provided in the paper). The aim of this paper is to provide a better complexity insight of the problem.

► Yang, X., Sikora, F., Blin, G., Hamel, S., Rizzi, R., and Aluru, S. (2012). An Algorithmic View on Multi-related-segments: a new unifying model for approximate common interval. In Agrawal, M., Cooper, S. B., and Li, A., editors, *9th annual conference on Theory and Applications of Models of Computation (TAMC)*, volume 7287 of *LNCS*, page 10pp

This article presented at the 9^{th} annual conference on Theory and Applications of Models of Computation (TAMC'12) considers and introduces a unifying model for the Approximate Common Intervals model. Recall that a set of genes that are proximately located on multiple chromosomes often implies their origin from the same ancestral genomic segment or their involvement in the same biological process. Among the numerous studies devoted to model and infer these gene sets, the recently introduced approximate common interval (ACI) models capture gene loss events in addition to the gene insertion, duplication and inversion events already incorporated by earlier models. However, the computational tractability of the corresponding problems remains open in most of the cases. In this contribution, we propose a unifying model for ACI, namely MULTI-RELATED-SEGMENTS (MRS), and demonstrate that capturing gene losses induces intractability in many cases. More precisely, we showed that modeling gene losses turns both search scenarios, where an ancestral gene set is given, and the general case, into computationally hard problems under the MRS model. The former is shown to be fixed parameter tractable, the latter to be **APX**-hard.

► Blin, G., Faye, D., and Stoye, J. (2010b). Finding Nested Common Intervals Efficiently. *Journal of Computational Biology*, 17(9):1183–1194

► Blin, G. and Stoye, J. (2009). Finding Nested Common Intervals Efficiently. In D., C. F. and István, M., editors, *7th RECOMB Satellite Workshop on Comparative Genomics (RECOMB-CG'09)*, volume 5817 of *Lecture Notes in Bioinformatics*, pages 59–69, Budapest, Hungary, Hongrie. Springer-Verlag

This article presented at the 7^{th} RECOMB Satellite Workshop on Comparative Genomics (RECOMB-CG'2009), Budapest, Hungary and extended for a journal version in Journal of Computational Biology tackles down the problem of efficiently finding gene clusters formalized by nested common intervals between two genomes represented either as permutations or as sequences. For permutations, we give several algorithms whose running time depends on the size of the actual

output rather than the output in the worst case. Indeed, we first provide a straightforward cubic time algorithm for finding all nested common intervals. We reduce this complexity by providing a quadratic time algorithm computing an irredundant output. We then show, by providing a third algorithm, that finding only the maximal nested common intervals can be done in linear time. Finally, we prove that finding approximate nested common intervals is fixed parameter tractable. For sequences, we provide solutions (modifications of previously defined algorithms and a new algorithm) for different variants of the problem, depending on the treatment one wants to apply to duplicated genes. This includes a polynomial-time algorithm for a variant implying a matching of the genes in the cluster, a setting that for other problems often leads to hardness.

► Blin, G., Fertin, G., Sikora, F., and Vialette, S. (2009b). The exemplar breakpoint distance for non-trivial genomes cannot be approximated. In Das, S. and Uehara, R., editors, *Proc. 3rd Annual Workshop on Algorithms and Computation (WALCOM'09), Kolkata, India*, volume 5431 of *Lecture Notes in Computer Science*, pages 357–368. Springer

This article presented at the 3rd Annual Workshop on Algorithms and Computation (WALCOM'2009), Kolkata, India considers the EXEMPLAR BREAKPOINT DISTANCE problem (or EBD, for short), which asks, given two genomes modeled by signed sequences of characters, to keep and match exactly one occurrence of each character in the two genomes (a process called exemplarization), so as to minimize the number of breakpoints of the resulting genomes. [56] showed that EBD is **NP**-complete. In this paper, we close the study of the approximation of EBD by showing that no approximation factor can be derived for EBD for non-trivial genomes — *i.e.* genomes that contain duplicated genes.

► Blin, G., Chauve, C., Fertin, G., Rizzi, R., and Vialette, S. (2007b). Comparing genomes with duplications: a computational complexity point of view. *ACM/IEEE Trans. Computational Biology and Bioinformatics*, 14(4):523–534

This article published in ACM/IEEE Transactions on Computational Biology and Bioinformatics focuses on the computational complexity of computing (dis)similarity measures between two genomes when they contain duplicated genes or genomic markers, a problem that happens frequently when comparing whole nuclear genomes. More precisely, it focuses on how to establish a one-to-one correspondence between genes of a pair of genomes, such that a given (dis)similarity measure for permutations is optimal. Considering two models to compute a one-to-one correspondence: the *exemplar* and the *matching* models, we show that for three (dis)similarity measures on permutations, namely the number of common intervals, the maximum adjacency disruption (MAD) number and the summed adjacency disruption (SAD) number, the problem of computing an optimal correspondence is **NP**-complete, and even **APX**-hard for the MAD and SAD numbers.

► Blin, G., Blais, E., Hermelin, D., Guillon, P., Blanchette, M., and El-Mabrouk, N. (2007a). Gene Maps Linearization using Genomic Rearrangement Distances. *Journal of Computational Biology*, 14(4):394–407

► Blin, G., Blais, E., Guillon, P., Blanchette, M., and El-Mabrouk, N. (2006). Inferring gene orders from gene maps using the breakpoint distance. In *Proc. 4th RECOMB Comparative Genomics Satellite*

Workshop (RECOMB-CG), Montréal, Canada, volume 4205 of *Lecture Notes in Bioinformatics*, pages 99–112

This article presented at the 4^{th} RECOMB Comparative Genomics Satellite Workshop (RECOMB-CG'2006), Montréal, Canada and extended for a journal version in Journal of Computational Biology considers a preliminary step to most comparative genomics studies; the so-called annotation of chromosomes as ordered sequences of genes. Indeed, different genetic mapping techniques often give rise to different maps with unequal gene content and sets of unordered neighboring genes. Only partial orders can thus be obtained from combining such maps. However, once a total order O is known for a given genome, it can be used as a reference to order genes of a closely related species characterized by a partial order P. Our goal is then to find a linearization of P that is as close as possible to O, according to a given genomic distance. We first prove **NP**-completeness results considering the breakpoint and the common interval distances. We then focuse on the breakpoint distance and gave a dynamic programming algorithm whose running time is exponential for general partial orders, but polynomial when the partial order is derived from a bounded number of genetic maps. A time-efficient greedy heuristic is then given for the general case and empirically shown to produce solutions within 10% of the optimal solution, on simulated data. Applications to the analysis of grass genomes were presented.

3 Biological Networks: Graphs

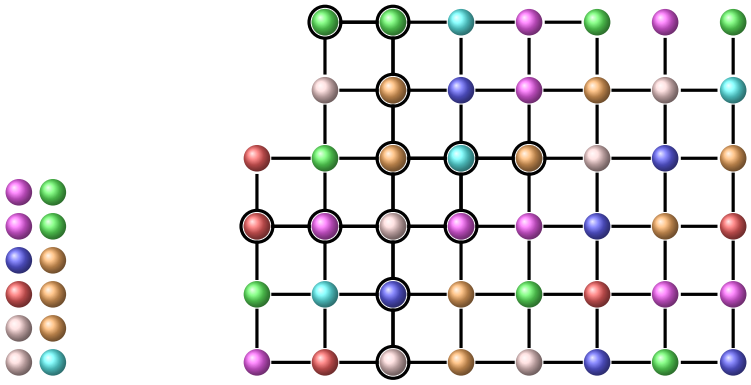
Contents

3.1 Introduction

Contrary to what was predicted years ago, the human genome project has highlighted that human complexity may not only rely on its genes (only 25 000 for human compared to the 30 000 and 45 000 for the mouse and the poplar respectively). This observation increased the interest in protein properties (*e.g.* their numbers, functions, complexity and interactions). Among other protein properties, the set of all their interactions for an organism, called *Protein-Protein Interactions* (PPI) networks, have attracted lot of interest. The number of reported interactions increases rapidly due to the use of various genome-scale screening techniques [74; 83; 123]. Unfortunately, acquiring such valuable resources is prone to high noise rate [74; 109].

A major issue of comparative analysis of PPI tries to determine to what extend proteins are conserved among species. Indeed, recent research suggests that proteins are functioning together into pathways (*i.e.*, a path in the interactions graph) or a structural complex (*i.e.*, an assembling of strongly connected proteins) and tend to evolve in correlated fashion – being preserved or eliminated in new species [106]. Therefore, it has became of foremost importance to identify PPI subnetworks that are similar to a given motif, where similarity is measured both in terms of protein-sequence and subnetwork topology conservation. This chapter is devoted to graph-based algorithmic aspects of pattern matching in PPI networks.

In our context, a PPI network is represented as a graph G where vertices are the proteins and edges are the interactions. In the classical view of PPI network querying, the pattern is also defined



a) b)

Figure 3.1: Instance of GRAPH MOTIF problem. a) a multiset M of colors, b) a graph G and a potential solution highlighted.

as a graph. Given a PPI network and a pattern, the problem is to find a subnetwork of the PPI network that is as similar as possible to the pattern, with respect to the initial topology. Similarity is measured both in terms of sequence similarity and graph topology conservation.

This problem, referred afterwards as GRAPH QUERY, considers topology-based motifs and is related to graph homomorphisms. The rationale for this research is that graph-homomorphisms do preserve adjacencies and hence are a natural choice for pattern matching problems in biological networks. However, as previously mentioned, interactions data are noisy and incomplete often inducing small differences that prevent finding some potential occurrences. Thus, in a more recent view of pattern matching in biological networks, researchers considered topology as of lesser importance and functionalities of network nodes forming the governing principle. According to this remark, [114] studied the following problem named GRAPH MOTIF: given a vertex-colored graph G and a multiset of colors M, does G have a connected subset of vertices whose multiset of colors equals M? In this context, the graph G represents the PPI network and the motif is completely defined by adding a color in M for each different requested proteins (see Figure 3.1). Once the motif is defined, a node v of G is colored by a color c of M if the protein represented by v is homologous to the protein represented by c (*e.g.*, according to some BLASTp analysis). If the protein represented by a node v is not homologous to any protein of the motif then v is not colored.

Note that, most of the results presented afterwards were obtained during the PhD thesis of Florian Sikora [118] that I co-supervised with Stéphane Vialette from 2008 to 2011.

3.2 Querying PPI Networks with topology

As previously mentioned, the GRAPH QUERY problem is clearly equivalent to the **NP**-complete SUBGRAPH HOMEOMORPHISM problem [73]. Recently, several techniques have been proposed to overcome the difficulty of this problem. By restricting the query to a path of length less than

five, [91] developped PATHBLAST. Unfortunately, PATHBLAST is an exponential-time algorithm which, worth to notice, allows flexibility. Indeed, PATHBLAST allows some mismatching between the pattern and its occurrence in the network (two consecutive mismatch is forbidden).

Later on, [117] proposed an alternative, called QPATH, for querying paths in a PPI network. The algorithm is based on the powerful color coding technique introduced by [2]. The use of this technique allowed the authors to define an FPT algorithm parameterized by the size of the query. In addition of being faster, QPATH deals with longer paths (~ 10) and allows more flexibility by considering a bounded number of mismatches.

By restricting the query to a tree, [107] proposed an algorithm that is restricted to forest PPI networks, *i.e.*, collection of trees. Finally, [65] developed QNET, a software to handle tree query in the general context of PPI networks. Of particular importance, [65] proposed an algorithm based on tree-decomposition for querying general graphs.

Let us present QNET which is the main reference in this field. QNET is an FPT algorithm for querying trees in a PPI network. The time complexity is $2^{\mathcal{O}(k)} m \ln(\frac{1}{\epsilon})$, where k is the number of proteins in the query, m the number of edges of the PPI network and $1-\epsilon$ the success probability (for any $\epsilon > 0$). As QPATH, QNET uses dynamic programming together with the color-coding technique. For querying graphs in a network, QNET uses, as a subroutine, an algorithm to query trees. To do so, it performs a tree decomposition (a formal definition of a tree decomposition can be found in [52]). Roughly speaking, it is a transformation of a graph into a tree, a tree node (or a bag) can contain several graph nodes. There exists several algorithms to perform such a transformation. The *treewidth* of a graph is the minimum (among all decompositions) of the cardinality of the largest bag minus one. Computing the treewidth is, however, **NP**-hard [8]. From this tree decomposition, the time complexity of QNET is $2^{\mathcal{O}(k)} n^{t+1} \ln(\frac{1}{\epsilon}))$ time, where k is the size of the query, n is the size of the PPI network, t is the treewidth of the query, and $1-\epsilon$ is the success probability (for any $\epsilon > 0$).

QNET is an algorithm for querying trees in a PPI network. A logical extension would be to query graphs. [65] provide a theoretical solution, without implementation and depending on the treewidth of the query. With F. Sikora and S. Vialette, we proposed PADA1 [45; 47] (Protein Alignment Dealing with grAphs) as an effective network querying algorithm extending QNET to more general query graphs.

As done in QNET, we defined PADA1 as a two-step procedure that first transforms the query graph into a tree and then uses that tree to effectively perform the query (allowing for insertions and deletions in the occurrence). QNET and PADA1 both rely on a tree-like query substructure. Despite that common base, the two algorithms use totally different approaches. Indeed, unlike QNET which is based on tree-decomposition, PADA1 exploits the fact that most query graphs have relatively small feedback vertex set (that is subset of vertices whose removal leads to a cycle-free graph) in practice. As the computation of the treewidth, finding a smallest feedback vertex set is a well-known **NP**-complete problem [73].

We propose a lossless transformation of a graph G into a tree (hence, one can reconstruct the graph starting from the tree) that iteratively finds a cycle C, duplicates (and stores) a node of C, and finally breaks cycle C by deleting one of its edges.

Let $F \subseteq V$ denote the set of all original nodes of $G = (V, E)$ that have been duplicated by this process. The cardinality of F turns out to be an important parameter since the overall time

complexity of PADA1 mostly depends on $|F|$ and not on the total number of duplications. Minimizing the cardinality of F is the well-known **NP**-complete FEEDBACK VERTEX SET problem [90]. In the current implementation of PADA1, we have implemented a "brute-force" algorithm for the FEEDBACK VERTEX SET problem. We did not considered more efficient approaches such as in [80; 122] since finding an occurrence of the constructed tree into the PPI network is definitively the most time-consuming part of our approach.

Indeed, in a second step, PADA1 consists in finding an occurrence (allowing insertions and deletions) of the constructed tree into the PPI network by combining random coloring and dynamic programming. The main difficulty here is to ensure to group process all the copies of a same vertex (which in the original instance correspond to a unique vertex). On the whole, the complexity of PADA1 is $O(mn^{|F|}N_{del}2^{O(k+N_{ins})}\log(\epsilon^{-1}))$ time, where k is the number of proteins in the query, m the number of edges of the PPI network, $1-\epsilon$ is the probability of success (for any $\epsilon > 0$), N_{ins} is the maximum number of insertions, N_{del} is the maximum number of deletions, and F is the feedback vertex set identified in the very first part of the algorithm. We showed in [45], that PADA1 was performing as well as QNet in practice and while the latter uses only trees, the former was able to query general graphs.

3.3 Querying PPI Networks without topology

From an algorithmic point of view, the GRAPH MOTIF problem introduced by [93] has been widely studied and depicted into numerous variations. The following section is based on an unformal overview of its complexity proposed by Florian Sikora during his PhD. In the following, $G = (V, E)$ will denote the vertex colored target network with $n = |V|$ and $m = |E|$. The motif M will be composed of k elements colored with c different colors. When $k = c$, M will be moreover denoted as *colorful*.

The problem was shown to be **NP**-complete by [93] as soon as the network is a tree. Later on, [69] proved that the problem remains **NP**-complete even for i) colorful motifs over networks represented as trees of maximum degree 3 and ii) motifs with 2 colors over networks represented as bipartite graph with maximum degree 4. On the bright side, the problem is solvable in $\mathcal{O}(n^{2cw+2})$ on graphs colored with constant treewidth w, once the number of colors c is also bounded. From a parameterized point of view, [69] proved both W[1]-hardness (parameterized by c) and membership to the FPT class – by providing an $\mathcal{O}(87^k.k.n^2)$ algorithm – when parameterized by the size of the motif.

[14] improved these results by designing an $\mathcal{O}(3^k.m)$ and an $\mathcal{O}(4.32^k.k^2.m)$ FPT algorithm for, colorful and general motifs, respectively. For colorful motifs, [55] designed an $\mathcal{O}(3^k.m.N_{ins})$ FPT algorithm allowing a multiset of colors in the occurrence.

[3] proved the problem to be polynomial for colorful motifs on caterpillars whereas **NP**-complete for i) colorful motifs over networks represented as rooted trees of height two, ii) colorful motifs over networks represented as trees even if a specific node (a root) is required in the solution and iii) colorful motifs over networks represented as graphs of diameter two.

[78] investigated the parameterized complexity of the problem and designed an $\mathcal{O}(2^k k^2 m)$ and an $\mathcal{O}(4^k k^2 m)$ time FPT algorithm using $\mathcal{O}(kn)$ space for colorful and general motifs respectively. [78] moreover generalized their algorithm for general motifs to handle deletions and r insertions in

$\mathcal{O}(4^k(k+r)^2m)$ time and $\mathcal{O}((k+r)n)$ space.

Despite the huge amount of theoretical results for the GRAPH MOTIF problem, to the best of our knowledge, there are only two implemented tools. [55] implemented a solution, called TORQUE, based on a combination of integer linear programming, dynamic programming and color coding. The first limitation of TORQUE is its weak possibilities of combination with others services since it is not standalone (it is a web service). Moreover, instead of providing all possible solutions, it can only provide one solution. Last but not least, it only deals with colorful motif.

In [46], with F. Sikora and S. Vialette, we implemented another tool for solving GRAPH MOTIF called GRAMOFONE as a Cytoscape (http://www.cytoscape.org/) plugin using Pseudo Boolean programming. It is worth noticing that our plugin also deals with some extensions of this problem. Indeed, due to the huge rate of noise in PPI Networks, exact match are often too restrictive, and hence one may allow deletions (*i.e.*, proteins which are in the motif but not in the solution). The resulting problem is MAX MOTIF, defined by [64], where a maximum sized connected occurence of M in G is requested.

[64] proved the problem to be **APX**-hard even for colorful motifs over networks represented as tree of maximum degree 3 where each color occurs at most twice. They proved the problem considering networks represented as trees to be not approximable within factor $2^{\log^{\delta} n}$, for any $\delta < 1$. From a parameterized point of view, they designed an $\mathcal{O}(k2^k n^3 \log n 2^{\mathcal{O}}(k)$ and an $\mathcal{O}(2^{5k}kn^2 \log^2 n 4^{\mathcal{O}}(k)$ FPT algorithm for general motifs over networks represented as trees and graphs respectively.

Similarly, the resulting subnetwork may contain protein insertions (*i.e.*, proteins which are in the solution but not in the motif) that help to get the connectivity of the result. Finally, since a protein can be homologous to more than one protein, a set of colors (rather than only one) can be assigned to any node of the network. The corresponding problem – the so-called LIST-COLORED GRAPH MOTIF problem – settled by [14], was shown to be solvable by an $\mathcal{O}(10.88^k.m)$ FPT algorithm for general motifs [14]. Later on, [78] provided an $\tilde{\mathcal{O}}(4^k k^2 m)$ time and $\tilde{\mathcal{O}}(kn)$ space FPT algorithm for general motifs.

GRAMOFONE was designed to consider both the MAX MOTIF and LIST-COLORED GRAPH MOTIF problems and thus can be seen as an integrated algorithmic toolbox to deal with the many flavors of the GRAPH MOTIF problem. TORQUE and GRAMOFONE perform more or less the same in terms of performances for moderate size tree motifs (GRAMOFONE is, however, not limited to trees). They also suffer from the same drawbacks: they are not able to deal with large motifs. However, GRAMOFONE is by far more scalable and is completely integrated into the cytoscape software (and hence can be easily used in combination with other cytoscape plugins). It is a challenging and important problem to improve GRAMOFONE so that it can tackle motifs of bigger size.

3.4 Presentation of papers

► Blin, G., Sikora, F., and Vialette, S. (2010e). GraMoFoNe: a cytoscape plugin for querying motifs without topology in protein-protein interactions networks. In Al-Mubaid, H., editor, *2nd International Conference on Bioinformatics and Computational Biology (BICoB-2010)*, pages 38–43, Honolulu,

USA. International Society for Computers and their Applications (ISCA)

This article presented at the 2nd International Conference on Bioinformatics and Computational Biology (BICoB'10), Honolulu, USA focuses on GRAMOFONE (http://igm.univ-mlv.fr/AlgoB/gramofone/). During the last decade, data on Protein-Protein Interactions (PPI) has increased in a huge manner. Searching for motifs in PPI Network has thus became a crucial problem to interpret this data. A large part of the literature is devoted to the query of motifs with a given topology. However, the biological data are, by now, so noisy (missing and erroneous information) that the topology of a motif can be unrelevant. Consequently, [93] defined a new problem, called GRAPH MOTIF, which consists in searching a multiset of colors in a vertex-colored graph. In this article, we present GRAMOFONE, a plugin to Cytoscape based on a Linear PseudoBoolean optimization solver which handles GRAPH MOTIF and some of its extensions.

► Blin, G., Sikora, F., and Vialette, S. (2010f). Querying Graphs in Protein-Protein Interactions Networks using Feedback Vertex Set. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 7(4):628–635. Special Issue-ISBRA 2009-Bioinformatics Research and Applications
► Blin, G., Sikora, F., and Vialette, S. (2009c). Querying Protein-Protein Interaction Networks. In Istrail, S., Pevzner, P., and Waterman, M., editors, *5th International Symposium on Bioinformatics Research and Applications (ISBRA'09)*, volume 5542 of *LNBI*, pages 52–62, Fort Lauderdale, FL, USA. Springer-Verlag

This article published in IEEE/ACM Transactions on Computational Biology and Bioinformatics and presented, in a shorter version, at the 5th International Symposium on Bioinformatics Research and Applications (ISBRA'09), Fort Lauderdale, USA describes our attempt to solve efficiently the GRAPH QUERY problem called PADA1. Recent techniques increase rapidly the amount of our knowledge on interactions between proteins. The interpretation of these new information depends on our ability to retrieve known sub-structures in the data, the Protein-Protein Interactions (PPI) networks. In an algorithmic point of view, it is an hard task since it often leads to **NP**-hard problems. To overcome this difficulty, many authors have provided tools for querying patterns with a restricted topology, *i.e.*, paths or trees in PPI networks. Such restriction leads to the development of fixed-parameter tractable (FPT) algorithms, which can be practicable for restricted sizes of queries. Unfortunately, GRAPH HOMOMORPHISM is a W[1]-hard problem, and hence, no FPT algorithm can be expected when patterns are in the shape of general graphs. However, [65] gave an algorithm (which is not implemented) to query graphs with a bounded treewidth in PPI networks (the treewidth of the query being involved in the time complexity). In this paper, we propose another algorithm for querying pattern in the shape of graphs, also based on dynamic programming and the color-coding technique. To transform graphs queries into trees without loss of informations, we use feedback vertex sets coupled to a node duplication mecanism. Hence, our algorithm is FPT for querying graphs with bounded feedback vertex sets. It gives an alternative to the treewidth parameter, which can be better or worst for a given query. We provide a python implementation which allows us to validate our implementation on real data. In particular, we retrieve some human queries in the shape of graphs into the fly PPI network.

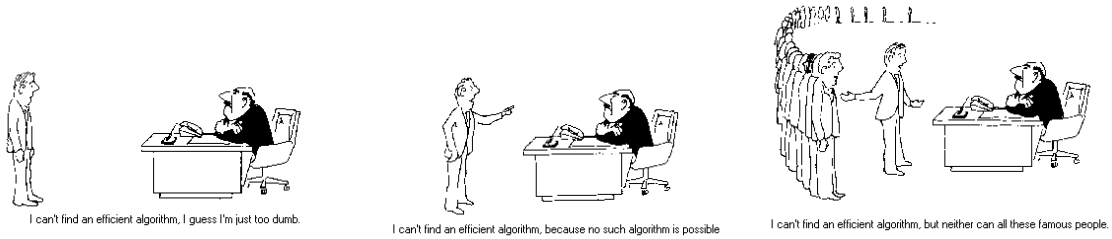
► Blin, G., Fertin, G., Mohamed-Babou, H., Rusu, I., Sikora, F., and Vialette, S. (2011b). Algorithmic

aspects of heterogeneous biological networks comparison. In W, W., X, Z., and D.-Z., D., editors, *5th Annual International Conference on Combinatorial Optimization and Applications (COCOA'11)*, volume 6831 of *Lecture Notes in Computer Science*, pages 272–286, Chine. Springer-Verlag

This article presented at the 5^{th} Annual International Conference on Combinatorial Optimization and Applications (COCOA'11), Zhangjiajie, China focuses on the NETWORK ALIGNMENT problem. Biological networks are commonly used to model molecular activity within the cell. Recent experimental studies have shown that the detection of conserved subnetworks across several networks, coming from different organisms, may allow the discovery of disease pathways and prediction of protein functions. There already exist automatic methods that allow to search for conserved subnetworks using networks alignment; unfortunately, these methods are limited to networks of same type, thus having the same graph representation. Towards overcoming this limitation, a unified framework for pairwise comparison and analysis of networks with different graph representations (in particular, a directed acyclic graph D and an undirected graph G over the same set of vertices) was introduced by Fertin et al. in 2010. We consider here a related problem called k-DAGCC: given a directed graph D and an undirected graph G on the same set V of vertices, and an integer k, does there exist sets of vertices $V_1, V_2, \dots V_{k'}$, $k' \leq k$ such that, for each $1 \leq i \leq k'$, (i) $D[V_i]$ is a DAG and (ii) $G[V_i]$ is connected ? Two variants of k-DAGCC are of interest: (a) the V_is must form a *partition* of V, or (b) the V_is must form a *cover* of V. We study the computational complexity of both variants of k-DAGCC and, depending on the constraints imposed on the input, provide several polynomial-time algorithms, hardness and inapproximability results.

Perspectives

The contributions provided during my PhD period were focusing on a binary question: does the studied problem is polynomial or not ? The answer that we usually provided with collaborators was either designing an exact and efficient algorithm when possible or, conversely, proving its **NP**-hardness. I have always been fascinated by the power of the **NP**-hardness. Indeed, as illustrated by the following famous drawing published by [73], rather than considering your failure to find an efficient solution to a given problem, you are able to prove that no one can find such one.



As soon as my PhD period, i was already interested on going further those kind of "negative" results since, as you may also have noticed yet, it seems that the difficulty of a problem is often proportional to the interest it causes (which somehow is quite frustrating as illustrated by the following drawing of D. Hermelin adapted from [73]).

Therefore I started to investigate both approximation and parameterized complexity. I have to admit that the second one always have my preference since it allows to solve hard problems exactly, rather than approximately, either by restricting attention to special cases, or by allowing some sort of confined exponential explosion in the running-time.

I would qualify the aim of my recent contributions as focusing on a more "practical" question: what makes a problem hard ? We try to provide solutions when possible. I pretty much appreciate the systematic procedure that we settled down with Stéphane Vialette that consist in investigating deeply the problems we are dealing with using a large complexity toolbox: Hardness reductions, Dynamic Programming, FPT theory, Color coding, Kernelization, Linear Programming. As I tried to emphasize in this thesis, a lot of interesting problems rely on well-defined combinatorial objects such as strings and graphs. In the future, I would like to pursue our systematic approach by focusing on those combinatorial objects and their intrinsic characteristics. I also wish to expand our studies on transversal problems that have applications in multiple domains. A first step towards this goal lies on the "Projets Exploratoires/Premier Soutien" (PEPS) CNRS "Traduction automatique et Génomique Comparative" that we obtained with S. Vialette and A.Allauzen (LIMSI) that tries to emphasize the links between comparative genomic and automatic traduction. We indeed found that our work on dotted-intervals (studied in the context of our ANR) may be used to help traduction process.

In the context of the "ANR Jeune Chercheur" project named BIRDS (2010-2014) for which I am the coordinator, a huge part of my research time is already devoted to specific subjects. In brief, this ambitious project is composed of three independent tasks: i) Algorithmic aspects of d-intervals, ii) Topology-free motifs in Biological Networks and iii) RadioTherapy. Rather than going into the details of those three subjects in this thesis, I will focus on the third one for which a PhD student (Paul Morel) that I co-supervise with S. Vialette just started. Considering our lack of collaboration on that field, in early 2010, I got contacts with a group in IOWA that was largely involved in the domain. This contact ended up in a 3 weeks visit in Xiaodong Wu research laboratory that allowed me to make contacts with Doctors in Oncology. Mainly, the outcome of this visit was the setting up of a co-direction for the PhD of Paul Morel – the ultimate goal being to be able to apply our results on that field to real data and with the feedback of practician. Even if it is quite an unusual experience for me, I am pretty confident that it will leads to a fruitful experience.

Finally, I would like to point out that we are in the process of starting a collaboration with the Henri Mondor Institute that is part of our "Pôles de Recherche et d'Enseignement Supérieur" – leading to yet another possibility of applying a bit more our theoretical research.

I would like to end this manuscript by presenting and briefly discussing two "extra" research topics I am particularly interested in nowadays (in my spare time) and on which I plan to work in the very near future. Regarding what I mentioned earlier, these topics have implication in bio-algorithmics but also in many other domains.

Colored Min-Cut

Recently, in collaboration with P. Bonizzoni, R. Dondi, R. Rizzi and F. Sikora, we investigated a challenging problem consisting in reconciling the gene and species trees with hypothetical gene duplications – referred as the MINIMUM DUPLICATION problem. This leads us to investigating a well-defined equivalent problem on graphs called MINIMUM CUT IN COLORED GRAPH problem. Given a set of colors C and a graph $G = (V, E)$ where any edge is colored with a color from C, find a minimal colored cut of G (that is a partition of V into two non-empty sets A and B such that the number of colors used by the edges having one end in A and the other in B is minimized). Despite

the huge amount of literature on MINIMUM CUT, almost nothing is known for the MINIMUM CUT IN COLORED GRAPH problem which seems to have numerous applications. By now, we "only" were able to design a randomized efficient solution.

Ranking aggregation

Considering any distance d, the OPTIMAL RANK AGGREGATION between a set R of m rankings (a ranking just corresponds to an ordering of elements) is a ranking – denoted as the optimal rank r_{OPT} – which minimizes $\sum_{r \in R} d(r_{OPT}, r)$. The problem have many applications in a variety of fields, and as much variations in the name. A couple of years ago, with M. Crochemore, S. Hamel, and S. Vialette, we focuses on the so-called Kendall-Tau distance. Roughly, this distance counts the number of pairwise disagreements between rankings. Once again, very few is known on this central problem. The OPTIMAL KENDALL-TAU RANK AGGREGATION problem is **NP**-complete even when considering 4 permutations [66]. But, nothing is known on the case of most interest for 3 permutations, which is in the context of phylogeny reconstruction often use as a subroutine. I would be interested in filling the knowledge gap on this distance but also considering other distances and generalizing to partial orders as done by [53].

Bibliography

[1] Alber, J., Gramm, J., Guo, J., and Niedermeier, R. (2002). Towards optimally solving the longest common subsequence problem for sequences with nested arc annotations in linear time. In Apostolico, A. and Takeda, M., editors, *Proc. 13th Annual Symposium on Combinatorial Pattern Matching (CPM), Fukuoka, Japan*, volume 2373 of *Lecture Notes in Computer Science*, pages 99–114. Springer.

[2] Alon, N., Yuster, R., and Zwick, U. (1995). Color coding. *Journal of the ACM*, 42(4):844–856.

[3] Ambalath, A. M., Balasundaram, R., H, C. R., Koppula, V., Misra, N., Philip, G., and Ramanujan, M. S. (2010). On the kernelization complexity of colorful motifs. Accepted at the 5th International Symposium on Parameterized and Exact Computation (IPEC 2010).

[4] Amir, A., Gasieniec, L., and Shalom, R. (2007). Improved approximate common interval. *Inf. Process. Lett.*, 103(4):142–149.

[6] Angibaud, S., Fertin, G., Rusu, I., Thévenin, A., and Vialette, S. (2007). A pseudo-boolean programming approach for computing the breakpoint distance between two genomes with duplicate genes. In *Proc. 5th RECOMB Comparative Genomics Satellite Workshop (RECOMB-CG)*, volume 4751 of *Lecture Notes in Bioinformatics*, pages 16–29. Springer.

[7] Angibaud, S., Fertin, G., Rusu, I., Thévenin, A., and Vialette, S. (2009). On the approximability of comparing genomes with duplicates. *J. Graph Algor. Appl.*, 13(1):19–53.

[8] Arnborg, S., Corneil, D., and Proskurowski, A. (1987). Complexity of finding embeddings in a k-tree. *Journal on Algebraic and Discrete Methods*, 8(2):277–284.

[9] Bergeron, A., Chauve, C., de Montgolfier, F., and Raffinot, M. (2008a). Computing common intervals of k permutations, with applications to modular decomposition of graphs. *SIAM J. Discret. Math.*, 22(3):1022–1039.

[10] Bergeron, A., Chauve, C., and Gingras, Y. (2008b). *Bioinformatics Algorithms: Techniques and Applications*, chapter 8, pages 177–202. Wiley & Sons, Inc.

[11] Bergeron, A., Corteel, S., and Raffinot, M. (2002a). The algorithmic of gene teams. In *Proceedings of WABI 2002*, volume 2452 of *Lecture Notes in Computer Science*, pages 464–476. Springer.

[12] Bergeron, A., Heber, S., and Stoye, J. (2002b). Common intervals and sorting by reversals: a marriage of necessity. *Bioinformatics*, 18(suppl 2):S54–S63.

[13] Bergeron, A. and Stoye, J. (2006). On the similarity of sets of permutations and its applications to genome comparison. *J. Comp. Biol.*, 13(7):1340–1354.

[14] Betzler, N., Fellows, M., Komusiewicz, C., and Niedermeier, R. (2008). Parameterized algorithms and hardness results for some graph motif problems. In *Proc. 19th Annual Symposium on Combinatorial Pattern Matching (CPM), Pisa, Italy*, volume 5029 of *Lecture Notes in Computer Science*, pages 31–43. Springer.

[15] Blin, G., Blais, E., Guillon, P., Blanchette, M., and El-Mabrouk, N. (2006). Inferring gene orders from gene maps using the breakpoint distance. In *Proc. 4th RECOMB Comparative Genomics Satellite Workshop (RECOMB-CG), Montréal, Canada*, volume 4205 of *Lecture Notes in Bioinformatics*, pages 99–112.

[16] Blin, G., Blais, E., Hermelin, D., Guillon, P., Blanchette, M., and El-Mabrouk, N. (2007a). Gene Maps Linearization using Genomic Rearrangement Distances. *Journal of Computational Biology*, 14(4):394–407.

[17] Blin, G., Bonizzoni, P., Dondi, R., Rizzi, R., and Sikora, F. (2012a). Complexity Insights of the Minimum Duplication Problem. In Bieliková, M., Friedrich, G., Gottlob, G., Katzenbeisser, S., and Turán, G., editors, *38th International Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM 2012)*, volume 7147 of *LNCS*, pages 153–164, Špindlerův Mlýn, Tchèque, République. Springer-Verlag.

[18] Blin, G., Bonizzoni, P., Dondi, R., and Sikora, F. (2012b). On the Parameterized Complexity of the Repetition Free Longest Common Subsequence Problem. *Information Processing Letters*.

[] Blin, G., Bulteau, L., Jiang, M., Tejada, P., and Vialette, S. (2012c). Longest common subsequences for bounded run lengths. In *Proc. 23th Annual Symposium on Combinatorial Pattern Matching (CPM), Helsinki, Finland*, Lecture Notes in Computer Science. Springer.

[19] Blin, G., Chauve, C., and Fertin, G. (2004a). The breakpoint distance for signed sequences. In *Proc. 1st Algorithms and Computational Methods for Biochemical and Evolutionary Networks (CompBioNets), Recife, Brazil*, pages 3–16. KCL publications.

[20] Blin, G., Chauve, C., and Fertin, G. (2005a). Genes order and phylogenetic reconstruction: Application to γ-proteobacteria. In *Proc. 3rd RECOMB Comparative Genomics Satellite Workshop*, volume 3678 of *LNBI*, pages 11–20.

[21] Blin, G., Chauve, C., Fertin, G., Rizzi, R., and Vialette, S. (2007b). Comparing genomes with duplications: a computational complexity point of view. *ACM/IEEE Trans. Computational Biology and Bioinformatics*, 14(4):523–534.

[22] Blin, G., Crochemore, M., Hamel, S., and Vialette, S. (2009a). Finding the median of three permutations under the Kendall-Tau distance. In *Proc. 7th annual international conference on Permutation Patterns, Firenze, Italy*. electronic version (6 pp).

[23] Blin, G., Crochemore, M., and Vialette, S. (2011a). *Algorithms in Computational Molecular Biology: Techniques, Approaches and Applications*, chapter Algorithmic Aspects of Arc-Annotated Sequences, pages 113–126. Wiley.

[24] Blin, G., Denise, A., Dulucq, S., Herrbach, C., and Touzet, H. (2010a). Alignment of RNA structures. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 7(2):309–322.

[25] Blin, G., Faye, D., and Stoye, J. (2010b). Finding Nested Common Intervals Efficiently. *Journal of Computational Biology*, 17(9):1183–1194.

[26] Blin, G., Fertin, G., Hermelin, D., and Vialette, S. (2005b). Fixed-parameter algorithms for protein similarity search under mrna structure constraints. In *Proc. 31st International Workshop International Workshop on Graph-Theoretic Concepts in Computer Science (WG), Metz, France*, volume 3787 of *Lecture Notes in Computer Science*, pages 271–282. Springer.

[27] Blin, G., Fertin, G., Hermelin, D., and Vialette, S. (2008). Fixed-parameter algorithms for protein similarity search under mrna structure constraints. *Journal of Discrete Algorithms*, 6(4):618–626.

[28] Blin, G., Fertin, G., Herry, G., and Vialette, S. (2007c). Comparing rna structures: towards an intermediate model between the edit and the lapcs problems. In Sagot, M.-F. and Telles Walter, M. E., editors, *1st Brazilian Symposium on Bioinformatics (BSB'07)*, volume 4643 of *Lecture Notes in Bioinformatics*, pages 101–112, Angra dos Reis, Brazil. Springer-Verlag.

[29] Blin, G., Fertin, G., Herry, G., and Vialette, S. (2007d). Comparing RNA structures: towards an intermediate model between the EDIT and the LAPCS problems. In Sagot, M.-F., Walter, W. T., and Maria, E., editors, *1st Brazilian Symposium on Bioinformatics (BSB), Angra dos Reis, Brazil*, volume 4643 of *Lecture Notes in Bioinformatics*, pages 101–112. Springer.

[30] Blin, G., Fertin, G., Mohamed-Babou, H., Rusu, I., Sikora, F., and Vialette, S. (2011b). Algorithmic aspects of heterogeneous biological networks comparison. In W, W., X, Z., and D.-Z., D., editors, *5th Annual International Conference on Combinatorial Optimization and Applications (COCOA'11)*, volume 6831 of *Lecture Notes in Computer Science*, pages 272–286, Chine. Springer-Verlag.

[31] Blin, G., Fertin, G., Rizzi, R., and Vialette, S. (2005c). What makes the arc-preserving subsequence problem hard ? *T. Comp. Sys. Biology*, 2:1–36.

[32] Blin, G., Fertin, G., Rizzi, R., and Vialette, S. (2005d). What makes the arc-preserving subsequence problem hard ? In *Proc Int. Workshop on Bioinformatics Research and Applications (IWBRA)*, volume 3515 of *Lecture Notes in Computer Science*, pages 860–868. Springer.

[33] Blin, G., Fertin, G., Rusu, I., and Sinoquet, C. (2007e). Extending the Hardness of RNA Secondary Structure Comparison. In Bo, C., Mike, P., and Guochuan, Z., editors, *1st intErnational Symposium on Combinatorics, Algorithms, Probabilistic and Experimental methodologies (ESCAPE'07)*, volume 4614 of *LNCS*, pages 140–151, Hangzhou, China, Chine. Springer-Verlag.

[34] Blin, G., Fertin, G., Sikora, F., and Vialette, S. (2009b). The exemplar breakpoint distance for non-trivial genomes cannot be approximated. In Das, S. and Uehara, R., editors, *Proc. 3rd Annual Workshop on Algorithms and Computation (WALCOM'09), Kolkata, India*, volume 5431 of *Lecture Notes in Computer Science*, pages 357–368. Springer.

[35] Blin, G., Fertin, G., and Vialette, S. (2004b). New results for the 2-interval pattern problem. In Sahinalp, S., Muthukrishnan, S., and Dogrusöz, U., editors, *Proc. 15th Annual Symposium on Combinatorial Pattern Matching (CPM), Istanbul,Turkey*, volume 3109 of *Lecture Notes in Computer Science*. Springer.

[36] Blin, G., Fertin, G., and Vialette, S. (2004c). New results for the 2-interval pattern problem. In *Proc. 15th Combinatorial Pattern Matching (CPM)*, volume 3109 of *Lecture Notes in Computer Science*, pages 311–322. Springer.

[37] Blin, G., Fertin, G., and Vialette, S. (2005e). What makes the arc-preserving subsequence problem hard ? *LNCS Transactions on Computational Systems Biology*, 2:1–36.

[38] Blin, G., Fertin, G., and Vialette, S. (2007f). Extracting constrained 2-interval subsets in 2-interval sets. *Theoretical Computer Science*, 385(1-3):241–263.

[39] Blin, G., Hamel, S., and Vialette, S. (2010c). Comparing RNA structures with biologically relevant operations cannot be done without strong combinatorial restrictions. In Rahman, M. S. and Fujita, S., editors, *4th Workshop on Algorithms and Computation (WALCOM'10)*, volume 5942 of *Lecture Notes in Computer Science*, pages 149–160, Dhaka, Bangladesh. Springer-Verlag.

[40] Blin, G. and Rizzi, R. (2005). Conserved intervals distance computation between non-trivial genomes. In *Proc. 11th Annual Int. Conference on Computing and Combinatorics (COCOON)*, volume 3595 of *Lecture Notes in Computer Science*, pages 22–31. Springer.

[41] Blin, G., Rizzi, R., Sikora, F., and Vialette, S. (2011c). Minimum Mosaic Inference of a Set of Recombinants. *International Journal of Foundations of Computer Science (IJFCS)*. To appear.

[42] Blin, G., Rizzi, R., Sikora, F., and Vialette, S. (2011d). Minimum Mosaic Inference of a Set of Recombinants. In Alex, P. and Taso, V., editors, *17th Computing: the Australasian Theory Symposium (CATS'11)*, volume 119 of *CRPIT*, pages 23–30, Perth, Australie. ACS.

[43] Blin, G., Rizzi, R., and Vialette, S. (2010d). A faster algorithm for finding minimum Tucker submatrices. In *6th Computability in Europe (CiE'10)*, volume 6158 of *Lecture Notes in Computer Science*, pages 69–77, Portugal. Springer.

[44] Blin, G., Rizzi, R., and Vialette, S. (2011e). A polynomial-time algorithm for finding minimal conflicting sets. In Kulikov, A. and Vereshchagin, N., editors, *Proc. 6th International Computer*

Science Symposium in Russia (CSR), volume 6651 of *Lecture Notes in Computer Science*, pages 373–384. Springer.

[45] Blin, G., Sikora, F., and Vialette, S. (2009c). Querying Protein-Protein Interaction Networks. In Istrail, S., Pevzner, P., and Waterman, M., editors, *5th International Symposium on Bioinformatics Research and Applications (ISBRA'09)*, volume 5542 of *LNBI*, pages 52–62, Fort Lauderdale, FL, USA. Springer-Verlag.

[46] Blin, G., Sikora, F., and Vialette, S. (2010e). GraMoFoNe: a cytoscape plugin for querying motifs without topology in protein-protein interactions networks. In Al-Mubaid, H., editor, *2nd International Conference on Bioinformatics and Computational Biology (BICoB-2010)*, pages 38–43, Honolulu, USA. International Society for Computers and their Applications (ISCA).

[47] Blin, G., Sikora, F., and Vialette, S. (2010f). Querying Graphs in Protein-Protein Interactions Networks using Feedback Vertex Set. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 7(4):628–635. Special Issue-ISBRA 2009-Bioinformatics Research and Applications.

[48] Blin, G. and Stoye, J. (2009). Finding Nested Common Intervals Efficiently. In D., C. F. and István, M., editors, *7th RECOMB Satellite Workshop on Comparative Genomics (RECOMB-CG'09)*, volume 5817 of *Lecture Notes in Bioinformatics*, pages 59–69, Budapest, Hungary, Hongrie. Springer-Verlag.

[49] Blin, G. and Touzet, H. (2006). How to Compare Arc-Annotated Sequences: The Alignment Hierarchy. In Crestani, F., Ferragina, P., and Sanderson, M., editors, *13th Symposium on String Processing and Information Retrieval (SPIRE'06)*, volume 4209 of *Lecture Notes in Computer Science*, pages 291–303, Glasgow, UK. Springer.

[50] Blin, G., Vialette, S., and Rizzi, R. (2012d). A faster algorithm for finding minimum Tucker submatrices. *Theory of Computing Systems*, page 10 pp.

[51] Böcker, S., Jahn, K., Mixtacki, J., and Stoye, J. (2009). Computation of median gene clusters. *J. Comput. Biol.*, 16(8):1085–1099.

[52] Bodlaender, H. (1993). A tourist guide through treewidth. *Acta Cybernetica*, 11:1–23.

[53] Boulakia, S. C., Denise, A., and Hamel, S. (2011). Using medians to generate consensus rankings for biological data. In Cushing, J. B., French, J. C., and Bowers, S., editors, *23rd International Scientific and Statistical Database Management (SSDBM 2011)*, volume 6809 of *Lecture Notes in Computer Science*, pages 73–90. Springer.

[54] Bourque, G., Yacef, Y., and El-Mabrouk, N. (2005). Maximizing synteny blocks to identify ancestral homologs. In *Proc. 3rd RECOMB Comparative Genomics Satellite Workshop*, volume 3678 of *LNBI*, pages 21–35.

[55] Bruckner, S., Hüffner, F., Karp, R., Shamir, R., and Sharan, R. (2009). Topology-free querying of protein interaction networks. In *Proc. 13th Annual International Conference on Computational Molecular Biology (RECOMB), Tucson, USA*, page 74. Springer.

[56] Bryant, D. (2000). The complexity of calculating exemplar distances. In Sankoff, D. and Nadeau, J., editors, *Comparative Genomics: Empirical and Analytical Approaches to Gene Order Dynamics, Map Alignment, and the Evolution of Gene Families*, volume 1, pages 207–212. Kluwer Academic Publisher.

[57] Chauve, C., Haus, U.-U., Stephen, T., and You, V. P. (2009). Minimal conflicting sets for the consecutive ones property in ancestral genome reconstruction. In Ciccarelli, F. and Miklós, I., editors, *RECOMB-CG 09*, volume 5817 of *Lecture Notes in Computer Science*, pages 48–58. Springer.

[58] Che, D., Li, G., and *et al.*, F. M. (2006). Detecting uber-operons in prokaryotic genomes. *Nucleic Acids Res*, 34(8):2418–2427.

[59] Chen, X., Zheng, J., Fu, Z., Nan, P., Zhong, Y., Lonardi, S., and Jiang, T. (2005). Assignment of orthologous genes via genome rearrangement. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 2(4):302–315.

[60] Chen, Z., Fowler, R. H., Fu, B., and Zhu, B. (2006a). Lower bounds on the approximation of the exemplar conserved interval distance problem of genomes. In Chen, D. Z. and Lee, D. T., editors, *COCOON*, volume 4112 of *Lecture Notes in Computer Science*, pages 245–254. Springer.

[61] Chen, Z., Fu, B., and Zhu, B. (2006b). The approximability of the exemplar breakpoint distance problem. In *2nd International Conference on Algorithmic Aspects in Information and Management (AAIM)*, volume 4041 of *Lecture Notes in Computer Science*, pages 291–302. Springer.

[62] Didier, G., Schmidt, T., Stoye, J., and Tsur, D. (2007). Character sets of strings. *J. Discr. Alg.*, 5(2):330–340.

[63] Dom, M., Guo, J., and Niedermeier, R. (2010). Approximation and fixed-parameter algorithms for consecutive ones submatrix problems. *Journal of Computer and System Sciences*, 76(3-4).

[64] Dondi, R., Fertin, G., and Vialette, S. (2009). Maximum motif problem in vertex-colored graphs. In Kucherov, G. and Ukkonen, E., editors, *Proc. 20th Annual Symposium on Combinatorial Pattern Matching (CPM'09), Lille, France*, volume 5577 of *Lecture Notes in Computer Science*, pages 221–235. Springer.

[65] Dost, B., Shlomi, T., Gupta, N., Ruppin, E., Bafna, V., and Sharan, R. (2007). QNet: A Tool for Querying Protein Interaction Networks. *RECOMB*, pages 1–15.

[66] Dwork, C., Kumar, R., Naor, M., and Sivakumar, D. (2001). Rank aggregation methods for the web. In *Proc. of the 10th international conference on World Wide Web (WWW), Hong Kong, Hong Kong*, pages 613–622.

[67] Evans, P. (1999a). *Algorithms and Complexity for Annotated Sequences Analysis*. PhD thesis, University of Victoria.

[68] Evans, P. A. (1999b). Finding common subsequences with arcs and pseudoknots. In Crochemore, M. and Paterson, M., editors, *Proc. 10th Annual Symposium Combinatorial Pattern Matching (CPM), Warwick University, UK*, volume 1645 of *Lecture Notes in Computer Science*, pages 270–280. Springer.

[69] Fellows, M., Fertin, G., Hermelin, D., and Vialette, S. (2007). Sharp tractability borderlines for finding connected motifs in vertex-colored graphs. In *Proc. 34th International Colloquium on Automata, Languages and Programming (ICALP), Wroclaw, Poland*, volume 4596 of *Lecture Notes in Computer Science*, pages 340–351. Springer.

[70] Felsenstein, J. (1988). Phylogenies from molecular sequences: Inference and reliability. *Ann. Review Genet.*, 22:521–565.

[71] Fitch, W. M. (2000). Homology – a personal view on some of the problems. *Trends Genet.*, 16:227–231.

[72] Fu, Z., Vacic, V., Zhong, Y., and Jiang, T. (2006). A parsimony approach to genome-wide ortholog assignment. In *in Research in Computational Molecular Biology, 10th Annual International Conference, RECOMB 2006*, pages 578–594. Springer.

[73] Garey, M. and Johnson, D. (1979). *Computers and Intractability: A guide to the theory of NP-completeness*. W.H. Freeman, San Francisco.

[74] Gavin, A., Boshe, M., et al. (2002). Functional organization of the yeast proteome by systematic analysis of protein complexes. *Nature*, 414(6868):141–147.

[75] Gramm, J., Guo, J., and Niedermeier, R. (2002). Pattern matching for arc-annotated sequences. In Agrawal, M. and Seth, A., editors, *Proc. 22nd Foundations of Software Technology and Theoretical Computer Science (FSTTCS), Kanpur, India*, Lecture Notes in Computer Science, pages 182–193. Springer.

[76] Gramm, J., Guo, J., and Niedermeier, R. (2006). Pattern matching for arc-annotated sequences. *ACM Transactions on Algorithms*, 2(1):44–65.

[77] Guignon, V., Chauve, C., and Hamel, S. (2005). An edit distance between rna stem-loops. In Consens, M. P. and Navarro, G., editors, *12th International Conference SPIRE*, volume 3772 of *Lecture Notes in Computer Science*, pages 335–347. Springer.

[78] Guillemot, S. and Sikora, F. (2010). Finding and counting vertex-colored subtrees. In Hlinený, P. and Kucera, A., editors, *35th International Symposium on Mathematical Foundations of Computer Science (MFCS'10)*, volume 6281 of *Lecture Notes in Computer Science*, pages 405–416, Brno, Czech Republic. Springer.

[79] Guo, J. (2002). Exact algorithms for the longest common subsequence problem for arc-annotated sequences. Master's thesis, Univeristy of Tübingen.

[80] Guo, J., Gramm, J., Hüffner, F., Niedermeier, R., and Wernicke, S. (2006). Compression-based fixed-parameter algorithms for feedback vertex set and edge bipartization. *Journal of Computer and System Sciences*, 72(8):1386–1396.

[81] He, X. and Goldwasser, M. H. (2005). Identifying conserved gene clusters in the presence of homology families. *J. Comp. Biol.*, 12(6):638–656.

[82] Heber, S. and Stoye, J. (2001). Finding all common intervals of k permutations. In Amir, A. and Landau, G., editors, *Proc. Annual Symposium on Combinatorial Pattern Matching (CPM), Jerusalem, Israel*, volume 2089 of *Lecture Notes in Computer Science*, pages 207–218. Springer.

[83] Ho, Y., Gruhler, A., et al. (2002). Systematic identification of protein complexes in Saccharomyces cerevisae by mass spectrometry. *Nature*, 415(6868):180–183.

[84] Hoberman, R. and Durand, D. (2005). The incompatible desiderata of gene cluster properties. In *Recomb-CG*, volume 3678 of *LNCS*, pages 73–87.

[85] Jahn, K. (2010). *Approximate Common Intervals Based Gene Cluster Models*. PhD thesis, Bielefeld University.

[86] Jiang, T., Lin, G., Ma, B., and Zhang, K. (2002). A general edit distance between RNA structures. *Journal of Computational Biology*, 9(2):371–388.

[87] Jiang, T., Lin, G., Ma, B., and Zhang, K. (2004). The longest common subsequence problem for arc-annotated sequences. *Journal of Dicrete Algorithms*, pages 257–270.

[88] Jiang, T., Lin, G.-H., Ma, B., and Zhang, K. (2000). The longest common subsequence problem for arc-annotated sequences. In Giancarlo, R. and Sankoff, D., editors, *Proc. 11th Annual Symposium on Combinatorial Pattern Matching (CPM), Montreal, Canada*, volume 1848 of *Lecture Notes in Computer Science*, pages 154–165. Springer.

[89] Jiang, T., Wang, L., and Zhang, K. (1995). Alignment of trees - an alternative to tree edit. *Theoretical Computer Science*, 143(1):137–148.

[90] Karp, R. (1972). Reducibility among combinatorial problems. In Thatcher, J. and Miller, R., editors, *Complexity of computer computations*, pages 85–103. Plenum Press, New York.

[91] Kelley, B., Sharan, R., Karp, R., Sittler, T., Root, D. E., Stockwell, B., and Ideker, T. (2003). Conserved pathways within bacteria and yeast as revealed by global protein network alignment. *Proceedings of the National Academy of Sciences*, 100(20):11394–11399.

[92] Klein, P. (1998). Computing the edit-distance between unrooted ordered trees. In Bilardi, G., Italiano, G., Pietracaprina, A., and Pucci, G., editors, *Proc. 6th European Symposium on Algorithms (ESA), Venice, Italy*, volume 1461 of *Lecture Notes in Computer Science*, pages 91–102. Springer.

[93] Lacroix, V., Fernandes, C., and Sagot, M.-F. (2006). Motif search in graphs: application to metabolic networks. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, 3(4):360–368.

[94] Landau, G. M., Parida, L., and Weimann, O. (2005). Gene proximity analysis across whole genomes via pq trees. *Journal of Computational Biology*, 12:1289–1306.

[95] Lin, G., Chen, Z.-Z., Jiang, T., and Wen, J. (2002a). The longest common subsequence problem for sequences with nested arc annotations. *Journal of Computer and System Sciences*, 65(3):465–480. Special issue on computational biology.

[96] Lin, G., Chen, Z.-Z., jiang, T., and Wen, J. (2002b). The longest common subsequence problem for sequences with nested arc annotations. *Journal of Computer and System Sciences*, 65:465–480.

[97] Lin, G., Ma, B., and Zhang, K. (2001). Edit distance between two rna structures. In *RECOMB*, pages 211–220.

[98] Ma, B., Wang, L., and Zhang, K. (2002). Computing similarity between RNA structures. *Theoretical Computer Sciences*, 276:111–132.

[99] Marron, M., Swenson, K. M., and Moret, B. M. E. (2003). Genomic distances under deletions and insertions. In *Theoretical Computer Science*, pages 537–547. Springer Verlag.

[100] Nadeau, J. H. and Taylor, B. A. (1984). Lengths of chromosomal segments conserved since divergence of man and mouse. *Proc Natl Acad Sci U S A*, 81(3):814–818.

[101] Nguyen, C. (2005). Algorithms for calculating exemplar distances. Technical report, National University of Singapore. Honours Year Project.

[102] Nguyen, C. T., Tay, Y. C., and Zhang, L. (2005). Divide-and-conquer approach for the exemplar breakpoint distance. *Bioinformatics*, 21:2171–2176.

[103] Ohno, S. (1970). *Evolution by gene duplication*. Springer-Verlag.

[104] Ouangraoua, A., Swenson, K., and Chauve, C. (2011). A 2-approximation for the minimum duplication speciation problem. *J Comput Biol*, 18(9):1041–1053.

[105] Pasek, S., Bergeron, A., and *et al.*, J. R. (2005). Identification of genomic features using microsyntenies of domains: domain teams. *Genome Res*, 15(6):867–874.

[106] Pellegrini, M., Marcotte, E., Thompson, M., Eisenberg, D., and Yeates, T. (1999). Assigning protein functions by comparative genome analysis: protein phylogenetic profiles. *PNAS*, 96(8):4285–4288.

[107] Pinter, R., Rokhlenko, O., Yeger-Lotem, E., and Ziv-Ukelson, M. (2005). Alignment of metabolic pathways. *Bioinformatics*, 21(16):3401–3408.

[108] Rahmann, S. and Klau, G. W. (2006). Integer linear programs for discovering approximate gene clusters. In *Proceedings of WABI 2006*, volume 4175 of *LNBI*, pages 298–309.

[109] Reguly, T., Breitkreutz, A., et al. (2006). Comprehensive curation and analysis of global interaction networks in saccharomyces cerevisiae. *Journal of Biology*.

[110] Sankoff, D. (1999). Genome rearrangement with gene families. *Bioinformatics*, 15(11):909–917.

[111] Sankoff, D. and Haque, L. (2005). Power boosts for cluster tests. In *Proc. 3rd RECOMB Comparative Genomics Satellite Workshop, Dublin, Ireland*, volume 3678 of *Lecture Notes in Bioinformatics*, pages 11–20.

[112] Sankoff, D. and Kruskal, B. (1983). *Time Warps, String Edits and Macromolecules: the Theory and Practice of Sequence Comparison*. Addison-Wesley.

[113] Sankoff, D., Leduc, G., Antoine, N., Paquin, B., Lang, B. F., and Cedergren, R. (1992). Gene order comparisons for phylogenetic inference: evolution of the mitochondrial genome. *Proceedings of the National Academy of Sciences*, 89(14):6575–6579.

[114] Schbath, S., Lacroix, V., and Sagot, M. (2009). Assessing the exceptionality of coloured motifs in networks. *EURASIP Journal on Bioinformatics and Systems Biology*, pages 1–9.

[115] Schmidt, T. and Stoye, J. (2004). Quadratic time algorithms for finding common intervals in two and more sequences. In *Proceedings of CPM 2004*, volume 3109 of *Lecture Notes in Computer Science*, pages 347–358. Springer.

[116] Shasha, D. and Zhang, K. (1989). Simple fast algorithms for the editing distance between trees and related problems. *SIAM Journal on Computing*, 18(6):1245–1262.

[117] Shlomi, T., Segal, D., Ruppin, E., and Sharan, R. (2006). QPath: a method for querying pathways in a protein-protein interaction network. *BMC Bioinformatics*, 7:199.

[118] Sikora, F. (2011). *Aspects algorithmiques de la comparaison d'éléments biologiques*. PhD thesis, Université Paris-Est Marne-la-vallée.

[119] Simillion, C., Vandepoele, K., and de Peer, Y. V. (2004). Recent developments in computational approaches for uncovering genomic homology. *Bioessays*, 26(11):1225–1235.

[120] Swenson, K. M., Marron, M., Earnest-DeYoung, J. V., and Moret, B. M. E. (2005). Approximating the true evolutionary distance between two genomes. In Demetrescu, C., Sedgewick, R., and Tamassia, R., editors, *ALENEX/ANALCO*, pages 121–129. SIAM.

[121] Tang, J. and Moret, B. (2003). Phylogenetic reconstruction from gene-rearrangement data with unequal gene content. In Dehne, F., Sack, J.-R., and Smid, M., editors, *Proc. 8-th International Workshop on Algorithms and Data Structures (WADS), Ottawa, Ontario, Canada*, volume 2748 of *Lecture Notes in Computer Science*, pages 37–46. Springer.

[122] Thomasse, S. (2009). A quadratic kernel for feedback vertex set. In *Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms*.

[123] Uetz, P., Giot, L., et al. (2000). A comprehensive analysis of protein-protein interactions in Saccharomyces cerevisae. *Nature*, 403(6770):623–627.

[124] Uno, T. and Yagiura, M. (2000a). Fast algorithms to enumerate all common intervals of two permutations. *Algorithmica*, 26(2).

[125] Uno, T. and Yagiura, M. (2000b). Fast algorithms to enumerate all common intervals of two permutations. *Algorithmica*, 26(2):290–309.

[126] Vialette, S. (2002). Pattern matching over 2-intervals sets. In Apostolico, A. and Takeda, M., editors, *Proc. 13th Annual Symposium Combinatorial Pattern Matching (CPM), Fukuoka, Japan*, volume 2373 of *Lecture Notes in Computer Science*, pages 53–63. Springer.

[127] Vialette, S. (2004). On the computational complexity of 2-interval pattern matching problems. *Theoretical Computer Science*, 312(2-3):223–249.

[128] Wang, B.-F. (2011). Output-sensitive algorithms for finding the nested common intervals of two general sequences. *IIEEE/ACM Trans Comput Biol Bioinform*.

[129] Yang, X. and Aluru, S. (2011). *Algorithms in Computational Molecular Biology: Techniques, Approaches and Applications*, chapter 32, pages 725–747. Wiley & Sons, Inc.

[130] Yang, X., Sikora, F., Blin, G., Hamel, S., Rizzi, R., and Aluru, S. (2012). An Algorithmic View on Multi-related-segments: a new unifying model for approximate common interval. In Agrawal, M., Cooper, S. B., and Li, A., editors, *9th annual conference on Theory and Applications of Models of Computation (TAMC)*, volume 7287 of *LNCS*, page 10pp.

[131] Zhang, M. and Leong, H. W. (2009). Gene team tree: A hierarchical representation of gene teams for all gap lengths. *J. Comp. Biol.*, 16(10):1383–1398.

[132] Zhu, B. (2009). Approximability and fixed-parameter tractability for the exemplar genomic distance problems. In *Proceedings of the 6th Annual Conference on Theory and Applications of Models of Computation*, TAMC '09, pages 71–80, Berlin, Heidelberg. Springer-Verlag.

Curriculum Vitæ

Civil Status

Born the 21th of December 1979 in St Germain en Laye (78) – French nationality

Contact details

Telephone:	01 60 95 77 49
Fax:	01 60 95 75 57
Email:	gblin@univ-mlv.fr
Office:	4B066
Webpage:	http://igm.univ-mlv.fr/~gblin
Professional Address:	Laboratoire d'Informatique Gaspard Monge Bât Copernic - Université de Marne la Vallée 5 Boulevard Descartes - Champs sur Marne 77454 Marne la Vallée Cedex 02

Marital status Single

Current status Associate Professor in the UPEMLV university (temporary full CNRS researcher – 2010/2011)

Scientific Training

- **2002-2005** – Doctorate at Université de Nantes, in Computer Science, defended the 17th of november 2005, with first-class honors.
  - **Title:** Combinatoire et Bio-informatique: Comparaison de structures d'ARN et Calcul de distances intergénomiques
  - **Place of defence:** UFR Sciences et Techniques, Université de Nantes
  - **Phd supervisor:** Irena RUSU, Full-Professor, Université de Nantes
  - **Phd coordinator:** Guillaume FERTIN, Full-Professor, Université de Nantes
  - **Laboratory:** Laboratoire d'Informatique de Nantes Atlantique (UMR CNRS 6241), Combinatoire et Bio-informatique (ComBi) team
  - **Grant:** Ministerial MENRT grant as a junior-lecturer

- ▷ **Jury:**
  - ▷ **President:** Guillaume FERTIN, Full-Professor, Université de Nantes
  - ▷ **Reviewers:**
    - · Marie-France SAGOT, Research Director, INRIA Rhône-Alpes
    - · Hélène TOUZET, Research Director CNRS (DR2), Université de Lille
  - ▷ **Examinators:**
    - · Romeo RIZZI, Full-Professor, Università degli Studi di Trento (Italy)
    - · Irena RUSU, Full-Professor, Université de Nantes
    - · Stéphane VIALETTE, Full-Researcher CNRS (CR1), Université de Marne la Vallée

► **2001-2002** – Master Thesis in Computer Science at Université de Nantes

- ▷ **Title:** Construction d'un environnement d'opérationalisation d'ontologies
- ▷ **Topic:** Artificial Intelligence
- ▷ **Date of defence:** September 2002
- ▷ **Place of defence:** UFR Sciences et Techniques, Université de Nantes
- ▷ **Scientific coordinators:**
  - ▷ Michel LECLERE, Associate-Professor, Université de Montpellier 2
  - ▷ Franky TRICHET, Associate-Professor, Université de Nantes
- ▷ **Laboratory:** Institut de Recherche en Informatique de Nantes (IRIN)

Professional Training

► **2011-2012** – 6 months of CRCT at Université de Marne la Vallée for handling the HAL project

► **2010-2011** – Full Temporary CNRS Researcher at Université de Marne la Vallée

► **From 2006** – Associate-Professor at Université de Marne la Vallée

► **2005-2006** – Assistant-Professor at Université de Marne la Vallée

► **2002-2005** – Junior-Lecturer - MENRT grant - Université de Nantes

► **2000-2002** – Temporary Lecturer at Université de Nantes

Scientific Duties

Local duties

From 2006, I am *co-organizing the weekly general seminar* of the Laboratoire d'Informatique Gaspard Monge. From 2006 to 2010, I have been one of the two organizers of the seminar. In 2009, I

suggested to spread the responsability of the organization to a member of each research team of the lab and, from then until 2011, is in charge of coordination. Moreover, in 2008, I *co-organized an interdisciplinary workgroup called "RécréAlgo"* which aims at puting together skills of researchers of different domains (but all linked to the theoretical computer science) around hard algorithmic problems. Note that the idea gave rise to a new workgroup in the context of the LABEX Bézout organized by Cyril Nicaud.

I am also, from 2006, *responsible of the bibliography* of the laboratory. In order to facilitate the update of this last, I have developed a tool allowing to any member of the laboratory to manage its bibliography entries. This tool also allows an automatic update of the web pages relative to the publications of the laboratory. As a consequence to this experience, I have been asked to be in charge of expending such tool at the scale of the Université Paris-Est Marne-la-vallée based on the tool HAL which leads to the actual HAL portal `http://hal-univ-mlv.archives-ouvertes.fr`.

Moreover, in collaboration with colleagues, we have implemented, in 2006, a *shared teaching area* allowing a collaborative work between teachers, reachable by our students and widely used.

Finally, I am *responsible of the web page* of the laboratory from 2009 and at the initiative of its new shape.

Committees

During the recruitment process of associate professors in 2009, I have been a *member of the "selection committee"* of the a) LIGM - Université de Marne-la-Vallée, b) LRI - IUT d'Orsay and c) LIFL - Faculté des sciences de Lille 1 (each for an associate-professor position). In 2010, I have, once again, been a member of the selection committee of the LRI - IUT d'Orsay. In 2011, I have been a member of the the selection committee of the a) LIGM - Université de Marne-la-Vallée and b) LRI - Faculté d'Orsay. In 2012, I have been a member of the selection committee of the LABRI - Université de Bordeaux. From January 2010, I am a *member of the permanent committee* and of the *laboratory board* of the LIGM.

I have been an external reviewer, among others, of the following journals: Algorithmica, Discrete Mathematics, Theoretical Computer Science, Journal of Computational Biology, Information Processing Letters, IEEE/ACM Transactions on Computational Biology and Bioinformatics, Journal of Combinatorial Optimization. I have also reviewed extended abstract papers for conferences such as CPM, WABI, SPIRE, STACS, PSB, RECOMB-CG, IWOCA.

In September 2007, I have co-organized some honorary days for Maxime Crochemore in the context of the *journées conjointes des groupes "Analyse de séquences" du GDR Bio-Informatique Moléculaire et "Combinatoire des mots, algorithmique du texte et du génome" du GDR Informatique Mathématique* in Marne-la-vallée (`http://www2.lifl.fr/SEQUOIA/Sequences/`). This event has attracted around one hundred researchers. In January and December 2011, I have again co-organized this event in Rennes and Lille respectively (`http://www.irisa.fr/symbiose/people/ppeterlongo/seqbi/`). I will be organizing this event once again in Marne-la-vallée in October 2012.

I have been part of the *program committee of RECOMB-CG* editions 2008, 2009 and 2010, and WABI 2012. In October 2008, I have been part of the *organizing committee* of the 2008 edition of RECOMB-CG which took place in Paris, France involving 120 participants and has been published as Lecture Notes in BioInformatics volume n°5267 proceedings. I will co-organize in October a

satellite workshop of SPIRE 2012.

International Mobility Development

I have settled (as a local responsible) two Erasmus exchange agreements: one with the Bielefeld University (Germany) which started in 2006 and has allowed a student of our first year of Master in Computer Science to obtain his Master Thesis in Bioinformatic from the Bielefeld University after a year in Germany and the other with the Brno University (Czech Republic) which started in 2009 and allowed two Czech phd students to visit our laboratory for a week in December 2010 and April 2012. A third agreement is nowadays in the preparation phase (with the Milano University (Italy)). Moreover, in 2010, I have applied and obtained an international postdoc mobility grant from the *Association Universitaire de la Francophonie* for a colleague (PhD David Celestin Faye) from the University Gaston-Berger of Saint-Louis (Senegal) which allowed him to make two short visits (3 months + 1 month) in my laboratory to work on a common project with Olivier Curé and myself. With Dr Faye, we are in the process of extending our collaborations at the laboratory and university levels by exploring Erasmus Mundus agreements.

Finally, via the process of "invited months", I have been able to provide some short terms visits to foreign collaborators to work with our team: Sylvie Hamel from Montréal, Canada (1 month in 2008), Jens Stoye from Bielefeld, Germany (1 month in 2008), Romeo Rizzi from Udine, Italy (1 month in 2009), Minghui Jiang from Utah, USA (1 month in 2011), Danny Hermelin from Saarbrücken, Germany (2 weeks in 2010), Xiao Yang from Iowa, USA (1 month in 2010), Riccardo Dondi from Milano, Italy (1 week in 2011), Sylvie Hamel from Montréal, Canada (1 week in 2011) and Xiaodong Wu from Iowa, USA (1 month in 2012).

Projects and Collaborations

I have been involved in numerous national and international projects that allowed me to strengthen my set of collaborators. Among those projects, we can mention

- The *Action Spécifique* CNRS - Département STIC "Nouveaux modèles et algorithmes de graphes pour la biologie" (2003-2004)
- The *Action Concertée Incitative* "Masse de Données NavGraphe" (2003-2006)
- The *Action Concertée Incitative* "Nouvelles Interfaces des Mathématiques pi-vert" (2005-2008)
- The working group "ARENA", based on the ACI IMPBio (2004-2007)
- The *programme blanc ANR* "BRASERO Biologically Relevant Algorithms and Softwares for Efficient RNA Structure Comparison" (2006-2010)
- A bilateral franco-italian project PAI Galileo n°08484VH (2005)
- A bilateral franco-quebec project from the *Commission Permanente de Coopération Franco-Québécoise* on "Structures conservées et duplications pour les réarrangements génomiques" (2005-2006)

- A *PEPS* CNRS: "Traduction Automatique et Génomique Comparative" (2010-2011)
- A *programme jeune chercheur ANR* "Biological networks, Radiotherapy and Structures" (2010-2014) for which I am the coordinator
- Member of the Laboratoire International Franco-Québécois de Recherche en Combinatoire (Laboratoire International Associé LIRCO)
- An "Investissements d'Avenir" program called ABS4NGS for "Solutions Algorithmiques, Bioinformatiques et Logicielles pour le Séquençage Haut Débit" (2012)

Some of those projects allowed me to make some short terms international visits: a month in 2005, 15 days in 2009 and 15 days in 2011 at Montréal, CANADA, 15 days in 2005 at Trento, ITALY, 10 days in 2009 at Lisboa, PORTUGAL, 15 days in 2010 at Udine/Milan, ITALY, 20 days in 2011 and 7 days in 2012 at Iowa, USA and 4 days in 2011 at Saarbrücken, GERMANY. These visits demonstrate my mobility skill (altough I am fully investigated in my teaching and scientific duties). It has been the occasion to meet most of my collaborators among which we can cite:

- Mathieu Blanchette - Montréal, CANADA
- Paola Bonizzoni - Milano, ITALY
- Cédric Chauve - Vancouver, CANADA
- Annie Chateau - Montréal, CANADA
- Maxime Crochemore - Paris, FRANCE / London, UK
- Riccardo Dondi - Milano, ITALY
- Nadia El Mabrouk - Montréal, CANADA
- David Faye - Saint Louis, SENEGAL
- Guillaume Fertin - Nantes, FRANCE
- Pierre Guillon - Nice, FRANCE
- Minghui Jiang - Utah, USA
- Sylvie Hamel - Montréal, CANADA
- Danny Hermelin - Haifa, ISRAEL / Saarbrücken, GERMANY
- Romeo Rizzi - Udine, ITALY
- Irena Rusu - Nantes, FRANCE
- Jens Stoye - Bielefeld, GERMANY

- Hélène Touzet - Lille, FRANCE
- Stéphane Vialette - Paris, FRANCE
- Xiaodong Wu - Iowa, USA
- Xiao Yang - Iowa, USA
- Michal Ziv Ukelson - Negev, ISRAEL

PhD Student Supervision

- Supervision (at 50% with Mr Stéphane Vialette) of a *PhD student* (MENRT grant) on biological networks – Florian Sikora between 2008 and 2011
- Supervision (at 50% with Mr Stéphane Vialette) of a *PhD student* (ANR grant) on radiotherapy – Paul Morel from September 2011

List of Publications

Complete list of my publications

Book Chapter

► Blin, G., Crochemore, M., and Vialette, S. (2011a). *Algorithms in Computational Molecular Biology: Techniques, Approaches and Applications*, chapter Algorithmic Aspects of Arc-Annotated Sequences, pages 113–126. Wiley

Journals

► Blin, G., Fertin, G., and Vialette, S. (2005e). What makes the arc-preserving subsequence problem hard ? *LNCS Transactions on Computational Systems Biology*, 2:1–36

► Blin, G., Blais, E., Hermelin, D., Guillon, P., Blanchette, M., and El-Mabrouk, N. (2007a). Gene Maps Linearization using Genomic Rearrangement Distances. *Journal of Computational Biology*, 14(4):394–407

► Blin, G., Chauve, C., Fertin, G., Rizzi, R., and Vialette, S. (2007b). Comparing genomes with duplications: a computational complexity point of view. *ACM/IEEE Trans. Computational Biology and Bioinformatics*, 14(4):523–534

► Blin, G., Fertin, G., and Vialette, S. (2007f). Extracting constrained 2-interval subsets in 2-interval sets. *Theoretical Computer Science*, 385(1-3):241–263

► Blin, G., Fertin, G., Hermelin, D., and Vialette, S. (2008). Fixed-parameter algorithms for protein similarity search under mrna structure constraints. *Journal of Discrete Algorithms*, 6(4):618–626

► Blin, G., Denise, A., Dulucq, S., Herrbach, C., and Touzet, H. (2010a). Alignment of RNA structures. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 7(2):309–322

► Blin, G., Faye, D., and Stoye, J. (2010b). Finding Nested Common Intervals Efficiently. *Journal of Computational Biology*, 17(9):1183–1194

► Blin, G., Sikora, F., and Vialette, S. (2010f). Querying Graphs in Protein-Protein Interactions Networks using Feedback Vertex Set. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 7(4):628–635. Special Issue-ISBRA 2009-Bioinformatics Research and Applications
► Blin, G., Rizzi, R., Sikora, F., and Vialette, S. (2011c). Minimum Mosaic Inference of a Set of Recombinants. *International Journal of Foundations of Computer Science (IJFCS)*. To appear
► Blin, G., Bonizzoni, P., Dondi, R., and Sikora, F. (2012b). On the Parameterized Complexity of the Repetition Free Longest Common Subsequence Problem. *Information Processing Letters*
► Blin, G., Vialette, S., and Rizzi, R. (2012d). A faster algorithm for finding minimum Tucker submatrices. *Theory of Computing Systems*, page 10 pp

In proceedings

► Blin, G., Chauve, C., and Fertin, G. (2004a). The breakpoint distance for signed sequences. In *Proc. 1st Algorithms and Computational Methods for Biochemical and Evolutionary Networks (CompBioNets), Recife, Brazil*, pages 3–16. KCL publications
► Blin, G., Fertin, G., and Vialette, S. (2004c). New results for the 2-interval pattern problem. In *Proc. 15th Combinatorial Pattern Matching (CPM)*, volume 3109 of *Lecture Notes in Computer Science*, pages 311–322. Springer
► Blin, G., Chauve, C., and Fertin, G. (2005a). Genes order and phylogenetic reconstruction: Application to γ-proteobacteria. In *Proc. 3rd RECOMB Comparative Genomics Satellite Workshop*, volume 3678 of *LNBI*, pages 11–20
► Blin, G., Fertin, G., Hermelin, D., and Vialette, S. (2005b). Fixed-parameter algorithms for protein similarity search under mrna structure constraints. In *Proc. 31st International Workshop International Workshop on Graph-Theoretic Concepts in Computer Science (WG), Metz, France*, volume 3787 of *Lecture Notes in Computer Science*, pages 271–282. Springer
► Blin, G. and Rizzi, R. (2005). Conserved intervals distance computation between non-trivial genomes. In *Proc. 11th Annual Int. Conference on Computing and Combinatorics (COCOON)*, volume 3595 of *Lecture Notes in Computer Science*, pages 22–31. Springer
► Blin, G., Fertin, G., Rizzi, R., and Vialette, S. (2005d). What makes the arc-preserving subsequence problem hard ? In *Proc Int. Workshop on Bioinformatics Research and Applications (IWBRA)*, volume 3515 of *Lecture Notes in Computer Science*, pages 860–868. Springer
► Blin, G. and Touzet, H. (2006). How to Compare Arc-Annotated Sequences: The Alignment Hierarchy. In Crestani, F., Ferragina, P., and Sanderson, M., editors, *13th Symposium on String Processing and Information Retrieval (SPIRE'06)*, volume 4209 of *Lecture Notes in Computer Science*, pages 291–303, Glasgow, UK. Springer
► Blin, G., Blais, E., Guillon, P., Blanchette, M., and El-Mabrouk, N. (2006). Inferring gene orders from gene maps using the breakpoint distance. In *Proc. 4th RECOMB Comparative Genomics Satellite Workshop (RECOMB-CG), Montréal, Canada*, volume 4205 of *Lecture Notes in Bioinformatics*, pages 99–112
► Blin, G., Fertin, G., Rusu, I., and Sinoquet, C. (2007e). Extending the Hardness of RNA Secondary Structure Comparison. In Bo, C., Mike, P., and Guochuan, Z., editors, *1st intErnational Symposium on Combinatorics, Algorithms, Probabilistic and Experimental methodologies (ESCAPE'07)*, volume 4614 of *LNCS*, pages 140–151, Hangzhou, China, Chine. Springer-Verlag
► Blin, G., Fertin, G., Herry, G., and Vialette, S. (2007d). Comparing RNA structures: towards an intermediate model between the EDIT and the LAPCS problems. In Sagot, M.-F., Walter, W. T., and

Maria, E., editors, *1st Brazilian Symposium on Bioinformatics (BSB), Angra dos Reis, Brazil*, volume 4643 of *Lecture Notes in Bioinformatics*, pages 101–112. Springer

► Blin, G., Sikora, F., and Vialette, S. (2009c). Querying Protein-Protein Interaction Networks. In Istrail, S., Pevzner, P., and Waterman, M., editors, *5th International Symposium on Bioinformatics Research and Applications (ISBRA'09)*, volume 5542 of *LNBI*, pages 52–62, Fort Lauderdale, FL, USA. Springer-Verlag

► Blin, G., Crochemore, M., Hamel, S., and Vialette, S. (2009a). Finding the median of three permutations under the Kendall-Tau distance. In *Proc. 7th annual international conference on Permutation Patterns, Firenze, Italy*. electronic version (6 pp)

► Blin, G., Fertin, G., Sikora, F., and Vialette, S. (2009b). The exemplar breakpoint distance for non-trivial genomes cannot be approximated. In Das, S. and Uehara, R., editors, *Proc. 3rd Annual Workshop on Algorithms and Computation (WALCOM'09), Kolkata, India*, volume 5431 of *Lecture Notes in Computer Science*, pages 357–368. Springer

► Blin, G. and Stoye, J. (2009). Finding Nested Common Intervals Efficiently. In D., C. F. and István, M., editors, *7th RECOMB Satellite Workshop on Comparative Genomics (RECOMB-CG'09)*, volume 5817 of *Lecture Notes in Bioinformatics*, pages 59–69, Budapest, Hungary, Hongrie. Springer-Verlag

► Blin, G., Sikora, F., and Vialette, S. (2010e). GraMoFoNe: a cytoscape plugin for querying motifs without topology in protein-protein interactions networks. In Al-Mubaid, H., editor, *2nd International Conference on Bioinformatics and Computational Biology (BICoB-2010)*, pages 38–43, Honolulu, USA. International Society for Computers and their Applications (ISCA)

► Blin, G., Hamel, S., and Vialette, S. (2010c). Comparing RNA structures with biologically relevant operations cannot be done without strong combinatorial restrictions. In Rahman, M. S. and Fujita, S., editors, *4th Workshop on Algorithms and Computation (WALCOM'10)*, volume 5942 of *Lecture Notes in Computer Science*, pages 149–160, Dhaka, Bangladesh. Springer-Verlag

► Blin, G., Rizzi, R., and Vialette, S. (2010d). A faster algorithm for finding minimum Tucker submatrices. In *6th Computability in Europe (CiE'10)*, volume 6158 of *Lecture Notes in Computer Science*, pages 69–77, Portugal. Springer

► Blin, G., Fertin, G., Mohamed-Babou, H., Rusu, I., Sikora, F., and Vialette, S. (2011b). Algorithmic aspects of heterogeneous biological networks comparison. In W, W., X, Z., and D.-Z., D., editors, *5th Annual International Conference on Combinatorial Optimization and Applications (COCOA'11)*, volume 6831 of *Lecture Notes in Computer Science*, pages 272–286, Chine. Springer-Verlag

► Blin, G., Rizzi, R., Sikora, F., and Vialette, S. (2011d). Minimum Mosaic Inference of a Set of Recombinants. In Alex, P. and Taso, V., editors, *17th Computing: the Australasian Theory Symposium (CATS'11)*, volume 119 of *CRPIT*, pages 23–30, Perth, Australie. ACS

► Blin, G., Rizzi, R., and Vialette, S. (2011e). A polynomial-time algorithm for finding minimal conflicting sets. In Kulikov, A. and Vereshchagin, N., editors, *Proc. 6th International Computer Science Symposium in Russia (CSR)*, volume 6651 of *Lecture Notes in Computer Science*, pages 373–384. Springer

► Blin, G., Bonizzoni, P., Dondi, R., Rizzi, R., and Sikora, F. (2012a). Complexity Insights of the Minimum Duplication Problem. In Bieliková, M., Friedrich, G., Gottlob, G., Katzenbeisser, S., and Turán, G., editors, *38th International Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM 2012)*, volume 7147 of *LNCS*, pages 153–164, Špindlerův Mlýn, Tchèque, République. Springer-Verlag

► Yang, X., Sikora, F., Blin, G., Hamel, S., Rizzi, R., and Aluru, S. (2012). An Algorithmic View on Multi-related-segments: a new unifying model for approximate common interval. In Agrawal, M., Cooper, S. B., and Li, A., editors, *9th annual conference on Theory and Applications of Models of Computation (TAMC)*, volume 7287 of *LNCS*, page 10pp

► Blin, G., Bulteau, L., Jiang, M., Tejada, P., and Vialette, S. (2012c). Longest common subsequences for bounded run lengths. In *Proc. 23th Annual Symposium on Combinatorial Pattern Matching (CPM), Helsinki, Finland*, Lecture Notes in Computer Science. Springer